Serums

HORIZON 2020

Project no. 826278

SERUMS

Research & Innovation Action (RIA)

SECURING MEDICAL DATA IN SMART-PATIENT HEALTHCARE SYSTEMS

Report on Refined Smart Health Centre System Software and Interoperability of Tools D6.2

Due date of deliverable: 31st December 2020

Start date of project: 1st January 2019

Type: Deliverable
WP number: WP6

Responsible Institution: University of St Andrews
Editor and editor's address: Juliana Bowles (jkfb@st-andrews.ac.uk)

Reviewers: Bram Elshof, Wanting Huang and Vladimir Janjic

Version 1.0

Project co-founded by the European Commission within the Horizon H2020 Programme		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

1 Release History

Release No.	**Dates**	**Author(s)**	**Release Description/Changes made**
V0.1	30/10/20 08/12/20 11/12/20 14/12/20	Thais Webber (USTAN) Agastya Silvina (USTAN) Guilherme Redeker (USTAN) Emma Morley (USTAN)	Status and detail on the SERUMS system integration and testing process, architectural aspects on the Smart Health Centre System (SHCS) and APIs, and WUI design. Added a brief study on the GDPR and local regulations. Added information and report from PoC2 execution at USTAN.
V0.2	15/12/20	Eduard Baranov (UCL) Thomas Given-Wilson (UCL)	Added sections concerning the modelling and properties verification
V0.3	22/12/20 24/12/20	Wanting Huang (ACC) Bram Elshof	Review on Blockchain module and integration information Review suggestions
V0.4	29/12/20	Juliana Bowles (USTAN)	Final review

2 SERUMS Consortium

Partner 1	**University of St Andrews**
Contact Person	Name: Juliana Bowles Email: jkfb@st-andrews.ac.uk
Partner 2	**Zuyderland Medisch Centrum**
Contact Person	Name: Larissa Haen-Jansen Email: la.jansen@zuyderland.nl
Partner 3	**Accenture B.V.**
Contact Person	Name: Bram Elshof Email: bram.elshof@accenture.com
Partner 4	**IBM Israel Science & Technology Ltd.**
Contact Person	Name: Michael Vinov Email: vinov@il.ibm.com
Partner 5	**Sopra-Steria**
Contact Person	Name: Andre Vermeulen Email: andreas.vermeulen@soprasteria.com
Partner 6	**Université Catholique de Louvain**
Contact Person	Name: Axel Legay Email: axel.legay@uclouvian.be
Partner 7	**Software Competence Centre Hagenberg**
Contact Person	Name: Michael Rossbory Email: michael.rossbory@scch.at
Partner 8	**University of Cyprus**
Contact Person	Andreas Pitsillides Email: andreas.pitsillides@ucy.ac.cy
Partner 9	**Fundació Clínic per a la Recerca Biomèdica**
Contact Person	Name: Santiago Iriso Email: siriso@clinic.cat
Partner 10	**University of Dundee**
Contact Person	Name: Vladimir Janjic Email: VJanjic001@dundee.ac.uk

Table of Contents

3 Executive Summary

Securing Medical Data in Smart Patient-Centric Healthcare Systems (SERUMS) is a research project supported by the European Commission (EC) under the Horizon 2020 program. This document is the second deliverable of Work Package 6: “Integration and Testing”. The leader of this work package is USTAN, with involvement from SOPRA, ACC, SCCH, IBM, UCL, SCCH, ZMC, and FCRB.

The purpose of this work package is to integrate the SERUMS technologies into a coherent Smart Health Centre System (SHCS) that will be used as a central access point to the different techniques developed over the course of the project.

We develop a front-end for the SHCS which considers different perspectives from which the data can be accessed, e.g. patient, specialist, and administration (T6.1). We integrate smart patient records, data analytics mechanisms, secure and privacy-preserving communication infrastructure and authentication mechanisms from WP2, WP3, WP4 and WP5 into this system (T6.2). Over the course of the integration, we will test and verify the interoperability of the SERUMS technologies on synthetic data that is produced by the data fabrication mechanisms from WP4 (T6.3). For interoperability issues identified throughout, we produce guidelines on how to address them. The testing and formal verification in this work package is done from a system development perspective. By contrast, WP7 deals with the evaluation of the use cases from a user’s perspective.

4 Introduction

4.1 Role of the Deliverable

This deliverable entitled "*Report on Refined Smart Health Centre System Software and Interoperability of Tools*" is the second deliverable of WP6. D6.2 reflects the refinement of the initial work reported in D6.1, which presented an initial design of the User Interface (UI) in the form of mock-ups and an architectural design proposal for the integration of SERUMS components within the Smart Health Centre System Software (SHCS). USTAN leads this task modifying the system developed in D6.1 to accommodate the necessary interoperation of the tools and techniques.

Continuing the work done in D6.1, we integrate the technologies developed in WP2–WP5 into the SHCS, analyse their interoperability, and identify any issues arising from their design and/or required privacy/security regulations such as GDPR (please refer to Appendix IV).

SOPRA, ACC, SCCH, IBM and UCY contribute to the SHCS by enabling the interoperability of their technologies from WP2, WP3, WP4 and WP5, while USTAN, ZMC and FCRB identify any issues coming from the specifics of their use cases. UCL contributes to the formal modelling and verification of the developed SHCS.

In summary, D6.2 updates the architectural design, describes the development phase, and reports the steps for technology integration, testing and formal verification, preparing the SERUMS integrated system for the subsequent tasks within WP6.

4.2 Relationship to Other SERUMS Deliverables

WP6 entitled "Integration and Testing" brings together work done across all WPs of the project. Overall, WP6 will consequently integrate smart patient records and authorisation schemes (WP2), data analytics mechanisms (WP3), secure and privacy-preserving communication infrastructure (WP4) and authentication mechanisms (WP5) into the SERUMS Smart Health Centre System (SHCS). Figure 1 shows the overview of the SERUMS work packages and their dependencies in the project execution.

Over the course of the integration, we test the interoperability of the SERUMS technologies on synthetic data that is produced by the data fabrication tool (WP4), and we will further investigate interoperability issues and produce guidelines on how to address them. The testing and formal verification to be performed in WP6 is done from a system's development perspective which includes testing and verifying the front-end and the functionality supported by the user interaction module (WUI - Web User Interface). By contrast, WP7 is concerned with the user evaluation of the three Use Cases - UCs – provided by USTAN, ZMC and FCRB. Deliverable D6.2 describes in detail the development work on the SHCS and it is associated with Milestone **MS11**.

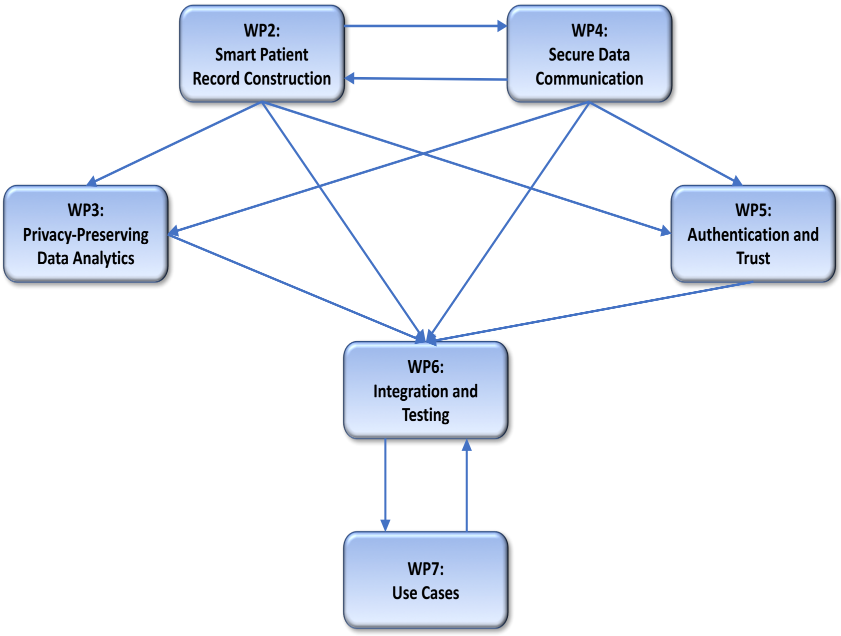


Figure 1 Overview of the SERUMS' WP Structure and Dependencies (PERT chart).

4.3 Structure of this Document

This document describes the SERUMS' toolchain design and integration [1,2], focusing on the SHCS's refined system architecture, recently added front-end functionality, and all development tasks (back-end) performed to date. Explicit mentions of contributions, from other WPs for the integration task, are made throughout this document in order to highlight the effort for the architectural design and SHCS/APIs development. The produced software artefacts during the design phase also incorporate the responsibilities of each partner as well as interfaces and steps required for a seamless SHCS integration [3].

The web user interface (WUI) considers the integration of the *Flexpass* authentication mechanism [4] (WP5) and associated front-end functionalities such as:

- access to the Smart Health Patient Record (SHPR) by users that are logged in to the system as patients (WP2);
- retrieval process of SPHR considering access rules and authorisation mechanisms in place (WP2);
- creation of access rules (WP2/WP5/WP6) to enable users (patients) to set access privileges (allow/deny actions) to other users (medical professionals);
- WUI translations to Dutch and Catalan.

In addition, the front-end included access to an Evaluation Questionnaire module (WP2/WP7) only deployed to be part of the Second Proof-of-Concept (PoC2) evaluation process (to be reported in D7.5).

Concerning GDPR and regional regulations applicable in three EU countries (cf. Appendix IV), information about *data subject rights*, *obligations of data controllers* (hospitals) and *joint-controllers* (in our case, the SERUMS project) were gathered. A summary on our major findings with respect to each Use Case (UC) is included. Other technical information concerning technology integration and the software development phase are also included in separate appendices of this document.

Formal modelling conducted to date on the SHCS system is based on timed automata, where the different components of the system are modelled as individual timed automata communicating through synchronous channels and reflecting both the design and architectural choices made for the system. Formal verification is done with the UPPAAL model checker [5]. This allows us to perform automated verification of system properties, describing all SHCS components as separate timed automata and a timed automata for representing user behaviour. Similarly, we can represent a model for an attacker of the system at different levels to investigate how the system behaves accordingly.

Finally, concluding remarks and further steps are outlined to be considered in the next WP6 deliverable (D6.3).

5 SERUMS Integrated System: architectural design

The SERUMS toolchain paper first published in 2019 [1] describes the overall process of accessing data across a distributed healthcare system. We have published another scientific paper in 2020 [2] concerning the SERUMS architectural design workflow, and how to address the different viewpoints for the architecture towards quality attributes. In the first SHCS design iteration, we explained that these different viewpoints are important artefacts to gain an understanding of the required components of the SHCS, as previously discussed in D6.1.

The second iteration of the SHCS design and integration have included refinements on the viewpoints (i.e., the designed artefacts) as well as a scheme of the system's architecture integration and deployment, using the proposed technologies. Figure 2 presents the SERUMS toolchain including its different components and their interactions. The core of SHCS is a centralised data lake that holds the distributed medical records information compiled in a unified format that allows the complex retrieval process. Note that, while the patient records are centralised as SPHR, the data in them may refer to databases distributed inside and outside hospital environments. These medical records contain all information about the patients: from static information such as date of birth, gender and contact information, to vital information such as weight, body mass index, allergies, to dynamic information about treatments and examinations. Some of the data for the records will be collected from within the healthcare system over trusted networks, while others may be collected from personal health monitoring devices, etc. Data sent over untrusted networks must be secured using data encryption mechanisms. In the context of this project, we are use patient synthetic data (obtained using IBM's Data Fabrication Platform), which will allow us to further demonstrate the efficiency of the SHCS.

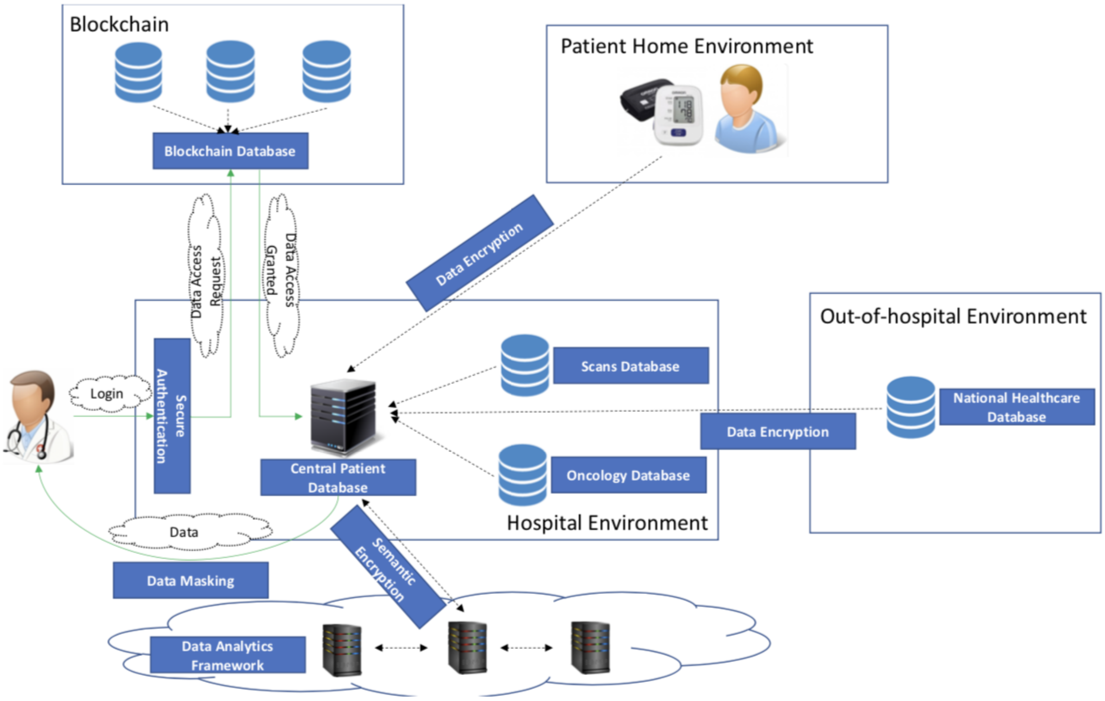


Figure 2 The overview of the SERUMS toolchain

When SERUMS users access patient data, they are first redirected to the Authentication client application. Once identified, the user is then redirected back to the SHCS client.

In the SERUMS project, our aim is to develop personalised and adaptive multi-factor user authentication schemes [4]. Once the user logs in, their access rights are checked using the Blockchain backend which is linked to a distributed Blockchain database. Different classes of users (e.g. patients, medical doctors, specialists, insurers, etc.) will have different levels of permissions in future versions of the system, including being compliant with GDPR[1] and other legal and ethical regulations.

For example, the patient has access to all available records the system can retrieve, while a specialist can only access parts of the record that are relevant to them. The Blockchain module ensures that only authorised agents can access the data, and depending on permissions, possibly only be part of the data. The Blockchain module contains all access rules and access transactions, and keeps a record of the data access history. Note, however, that no actual patient data is stored in the blockchain. Once the user is authenticated and the access rights are checked, the requested data from the smart patient health records in the Data Lake is sent back to the user. The access transaction itself is stored in the Blockchain database. Following, we decompose the architecture of the whole system into several viewpoints that clarify specific aspects for the integration task. In this report we refine the designed viewpoints presented in D6.1 adding details and modifications originated in the integration task, as well as detail on the WPs responsibilities within the development phase.

5.1 Context Viewpoint

The Context Viewpoint presents the context for all the components in the SHCS. It shows how the different users will interact with the Web User Interface (WUI) of the SHCS as the main access point and authentication web client for authentication. The WUI connects to the integrated modules, the SERUMS API, which forwards all the requests to the internal components, including the Authentication module (WP5), the Blockchain module (WP2/WP5), and the Data Lake module (WP2). The user does not have any direct interaction with the IBM data synthesizer module (WP4) or the Privacy-preserving module (WP3), because these functionalities are reserved for the research and development team only. However, there is an existing connection between the Data Lake and the generated patient synthetic data, which will allow us to further demonstrate and evaluate the SHCS operation, features and usage.

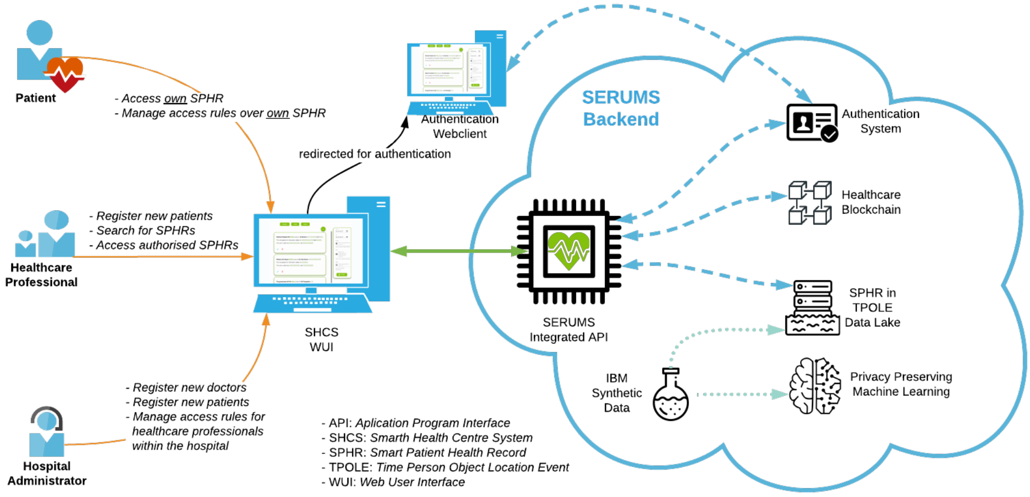


Figure 3 Context Viewpoint

[1] Information on GDPR can be found at https://gdpr-info.eu/

5.2 Functional Viewpoint

The Functional Viewpoint details the functionalities of each module, and how the integration modules (SHCS and SERUMS API) connect to the other modules via these APIs. This architectural decision was made to enforce decoupling between components, reducing the chances of systemic failures, while easing the integration task. The functionalities are presented according to the requirements assessment step performed by the WP7 partner for each Use Case (UC) - USTAN, ZMC and FCRB.

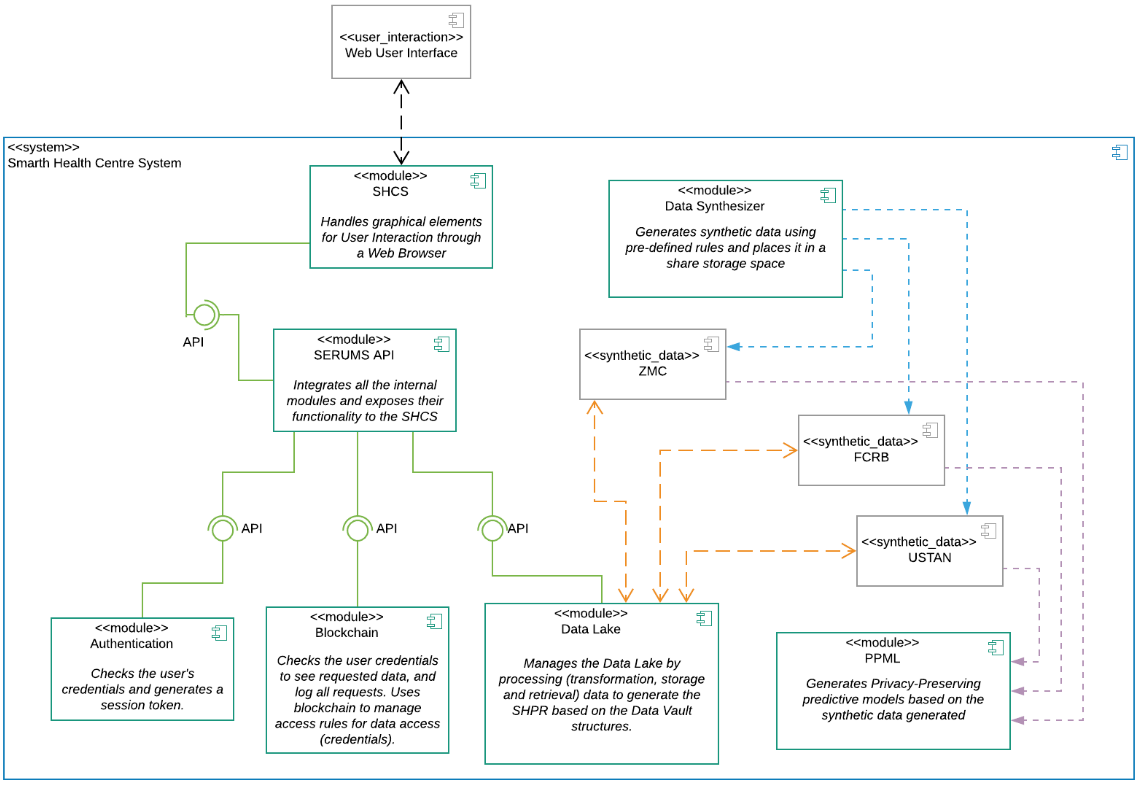

<<user_interaction>>
Web User Interface
<<system>>
Smarth Health Centre System
<<module>>
SHCS
Handles graphical elements for User Interaction through a Web Browser
<<module>>
Data Synthesizer
Generates synthetic data using pre-defined rules and places it in a share storage space
API
<<module>>
SERUMS API
Integrates all the internal modules and exposes their functionality to the SHCS
<<synthetic_data>>
ZMC
<<synthetic_data>>
FCRB
<<synthetic_data>>
USTAN
API
API
API
<<module>>
Authentication
Checks the user's credentials and generates a session token.
<<module>>
Blockchain
Checks the user credentials to see requested data, and log all requests. Uses blockchain to manage access rules for data access (credentials).
<<module>>
Data Lake
Manages the Data Lake by processing (transformation, storage and retrieval) data to generate the SHPR based on the Data Vault structures.
<<module>>
PPML
Generates Privacy-Preserving predictive models based on the synthetic data generated

Figure 4 Functional Viewpoint

5.3 Interaction Viewpoint

The Interaction Viewpoint describes the APIs to be implemented by the main components of the SHCS. In the diagram (Figure 5) the viewpoint depicts how the (User) Authentication module can provide mechanisms to register new users, activate registered users, and to authenticate active users within the SERUMS toolchain.

The Blockchain module will further handle the creation, deletion and update of access rules, and the log of events occurred in the toolchain. Finally, the Data Lake module will provide a service to retrieve the Smart Patient Health Record (SPHR) at each request, which will automatically integrate synthetic data in the future into the proposed SPHR unified format. The current PoC2 version integrates only one user (patient) data for each use case (UC).

The SERUMS API works as a bridge between the SHCS and the other APIs. If the modules of the toolchain will be deployed in different locations, it would be the main features of SERUMS API to connect and access these modules from different locations.

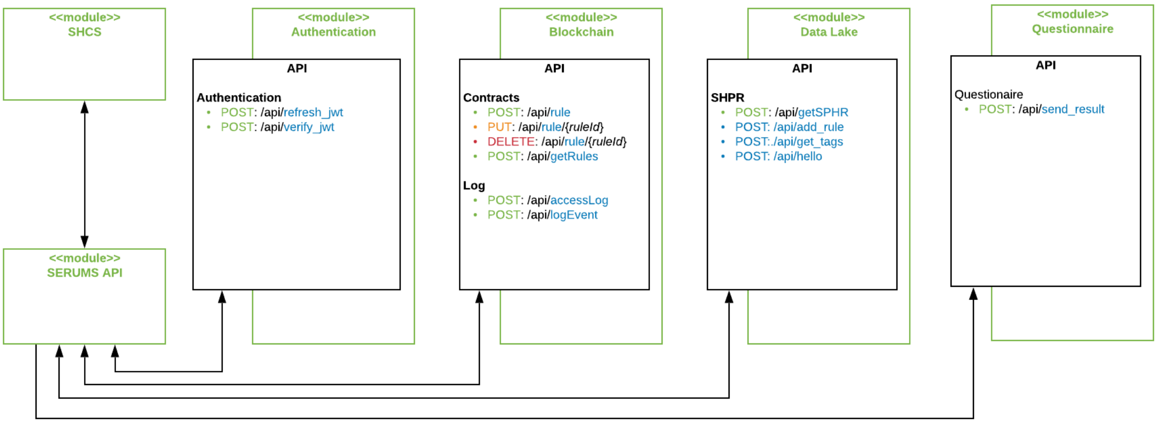


Figure 5 Interaction Viewpoint

5.4 Deployment Viewpoint

The SERUMS toolchain [1] is composed of different modules or subsystems that can live independently. However, for the SHCS to function properly, all the modules must be able to communicate amongst them.

We have leveraged the features of the Docker[2] technology to produce a seamless integration (Figure 6). In this way all subsystems (modules) will be containerized and deployed in a single managed server. With this architecture structure, all communications between modules happen internally across the same network. The use of Docker containers is common in projects where replication is crucial, such as the case of SERUMS. All the technical partners can work independently in their modules using different platforms, programming languages, operative systems, etc., and the integration module will be able to interact with them in a homogenized way.

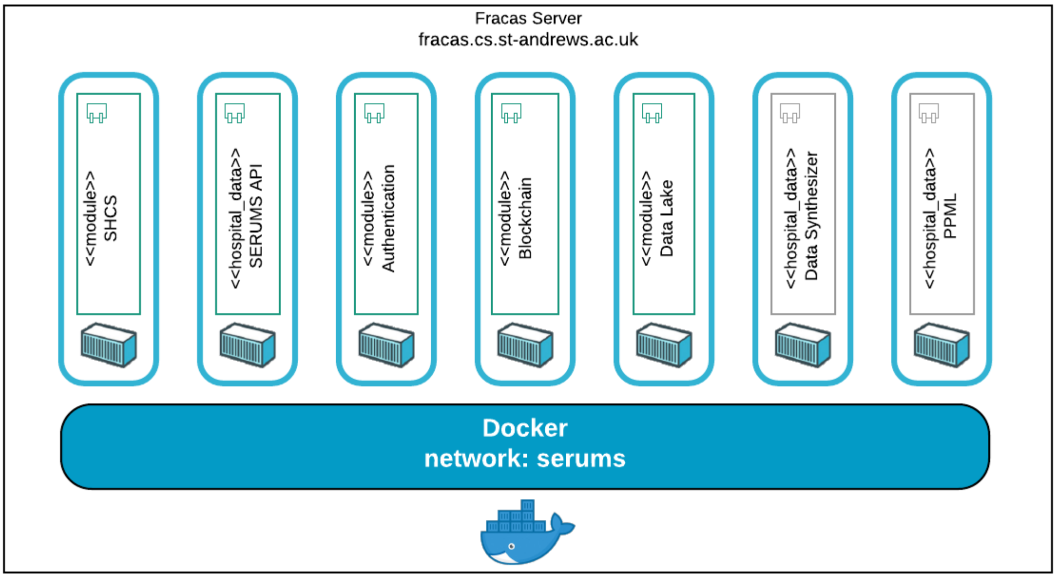


Figure 6 Deployment Viewpoint

[2] Docker technology information can be found at https://www.docker.com/

5.5 Sequence Diagram

The Sequence Diagram in Figure 7 shows the steps required, e.g., for a doctor to retrieve the SPHR of a given patient. We assume that both the patient and the doctor are already registered in the system, but the doctor does not have access rules created to the patient's SPHR yet. Then the first part of the use case required the patient to login to the system and create access rules to allow the doctor to retrieve his/her SPHR. In the second part, the doctor must login as well, and then request the retrieval of the patient's SPHR. This request will go all the way through the Blockchain module checking user's privileges, and to the Data Lake to retrieve the SPHR information with the specific synthetic data this particular doctor is authorised to see.

In the current PoC2 version, patients as users are able to retrieve their own SPHR and create access rules. Doctors as users will be integrated in future versions of the SCHS, allowing us to test the outputs of the Data Lake retrieval process combined with Blockchain authorisation mechanism.

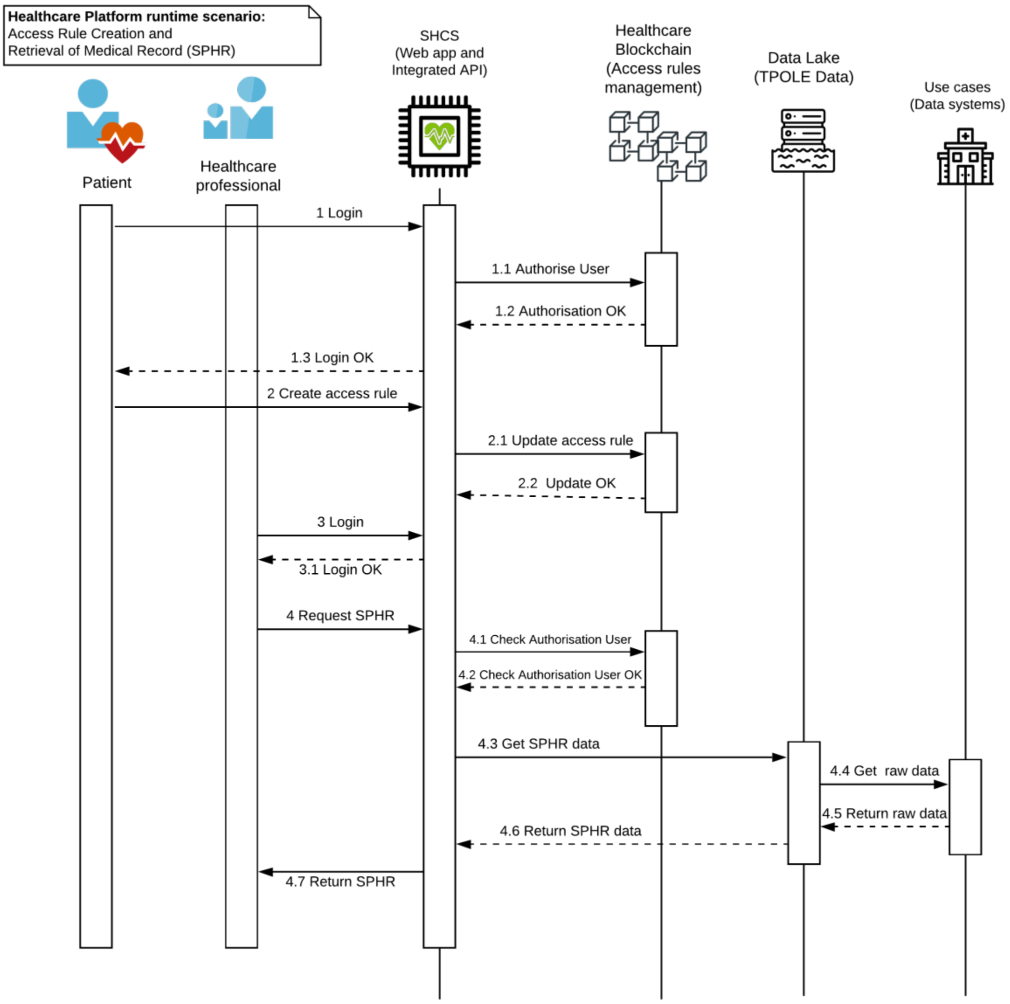


Figure 7 Sequence Diagram (Access rules creation and retrieval request process of SPHR)

5.6 Information Flow Viewpoint

The Information Flow Viewpoint (Figure 8, also shown in larger resolution on Appendix I) shows what information flows between each component (module) along the use case presented in Figure 7 as a sequence diagram from the perspective of a user. First the SHCS requests an access token to the authentication module using the user credentials, if authenticated, this token will enable the user to continue with the navigation. Then the user (e.g., a healthcare professional, a doctor) can request a patient's SPHR, where this request will have to be authorised by the Blockchain module, issuing an authorization token that will be forwarded to the Data Lake module in order to filter the patient's information to be shown to the user.

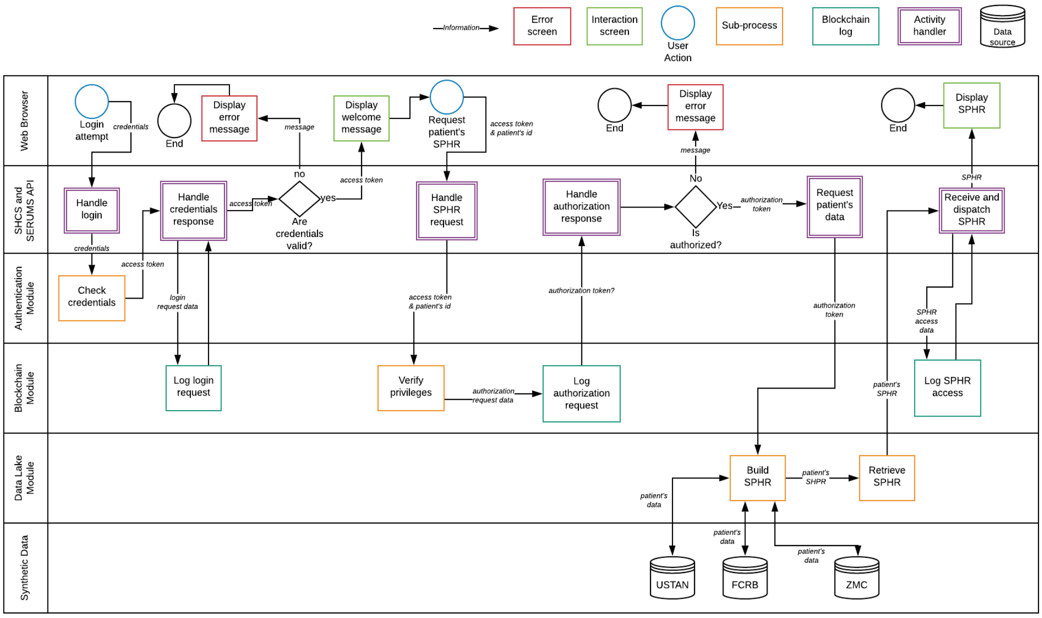


Figure 8 Information Flow Viewpoint

These above mentioned viewpoints are the basis of the SHCS architecture, i.e. the foundation for the system integration. In the following sections we present the detailed information on the development phase as well as detail on the Web User Interface (WUI) integration.

6 SERUMS Integrated System: development phase

The SHCS is a client application written in JavaScript and developed with React framework. Currently, the system integrates the following modules: the Authentication system (WP5), the SPHR (Smart Patient Health Records) - Data Lake module (WP2), the Access Rules Creation module - joint development among WP6 (SHCS integration), WP2 (Data Lake) and WP2/WP5 (Blockchain), and the Evaluation Questionnaire module - joint development among WP2 (coding), WP6 (SHCS integration) and WP7 (Evaluation Questionnaire creation and further translations to Dutch and Catalan).

6.1 Authentication module integration

The SERUMS system does not locally store user data (i.e., cookies) in this particular PoC2 version because it would require additional patients' consent form implemented in the system.

The first action is to redirect users (e.g., patients) to the authentication web-page https://flexpass.SERUMS .cs.st-andrews.ac.uk/web_app/index.html (*Flexpass* system [4], developed by University of Cyprus/WP5).

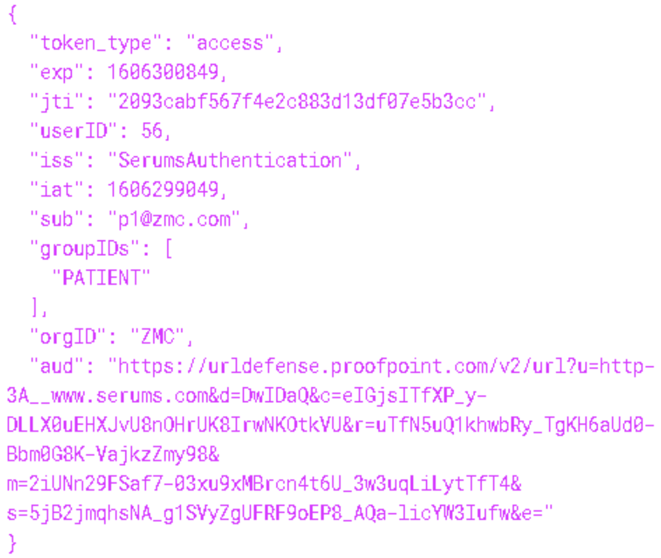
In order to do that, the system implements an *AppWrapper* in its initialisation, a wrapping module to handle the authentication as follows:

1. At first, *AppWrapper* redirect the user to the Authentication client;
2. Once the Authentication client redirects back with JWT (tokens) in the URL, the *AppWrapper* will parse the token and perform verification or refresh token.
3. If the token is verified as 'valid', the *AppWrapper* redirects the user to the welcome page.

We use JSON Web Tokens[3] (JWT), which is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. The information can be verified and trusted because it is digitally signed. The most common scenario for using JWT is on authorisation processes. JWT tokens are created by the authentication module in the SERUMS system. Once the user is logged in, each subsequent request will include the JWT, allowing the user to access routes, services, and resources that are permitted with that token.

JWT created by the authentication module in the SERUMS system retrieves the patient information as a JSON file such as the code excerpt shown in Figure 9.

[3] Information about JSON Web tokens can be found at https://jwt.io/introduction/



```
{
  "token_type": "access",
  "exp": 1606300849,
  "jti": "2093cabf567f4e2c883d13df07e5b3cc",
  "userID": 56,
  "iss": "SerumsAuthentication",
  "iat": 1606299049,
  "sub": "p1@zmc.com",
  "groupIDs": [
    "PATIENT"
  ],
  "orgID": "ZMC",
  "aud": "https://urldefense.proofpoint.com/v2/url?u=http-
3A__www.serums.com&d=DwIDaQ&c=eIGjsITfXP_y-
DLLX0uEHXJvU8nOHrUK8IrwNKOtkVU&r=uTfN5uQ1khwbRy_TgKH6aUd0-
Bbm0G8K-VajkzZmy98&
m=2iUNn29FSaf7-03xu9xMBrcn4t6U_3w3uqLiLytTfT4&
s=5jB2jmqhsNA_g1SVyZgUFRF9oEP8_AQa-licYW3Iufw&e="
}
```

Figure 9 JSON Web tokens implementation code excerpt

There are two different token types: *JWT access* and *JWT refresh*. *JWT access* (Figure 9) is used for retrieving the data from several modules (e.g., Data Lake, Blockchain) and *JWT Refresh* is used when the token expires.

In a compact JWT structure it consists of three parts, which are: header, payload and signature. Figure 9 shows the payload, which contains the claims. Claims are statements, e.g., about the user, and additional data. The predefined claims are basically: iss (issuer), exp (expiration time), sub (subject), aud (audience), and others.

In the SERUMS integration case, the predefined claims are the following:

- `exp`: states the expiration date of the tokens in timestamp format;
- `userId`: is the SERUMS user ID;
- `sub`: is the email used for login using the authentication model;
- `orgId`: is the base organisation for each patient. This information defines the default language for the SERUMS SHCS client;
- `groupIDs`: defines the user identity. After a user finishes authentication, they are redirected to the specific page based on their `groupID`. On this system version, we only implemented several modules for users that identify as PATIENT.

6.2 Data Lake module integration

The Smart Patient Health Records (SPHR) module (WP2) retrieves patients' records from the Data Lake integrating them for displaying the data in the SERUMS WUI. The components within SHCS are implemented using React Hook[4], which allows developers to use state and other React features without writing a class. With Hooks, we can extract stateful logic from a component so it can be tested independently and reused for the future test implementation.

The main building blocks for this component in the SHCS are the following:

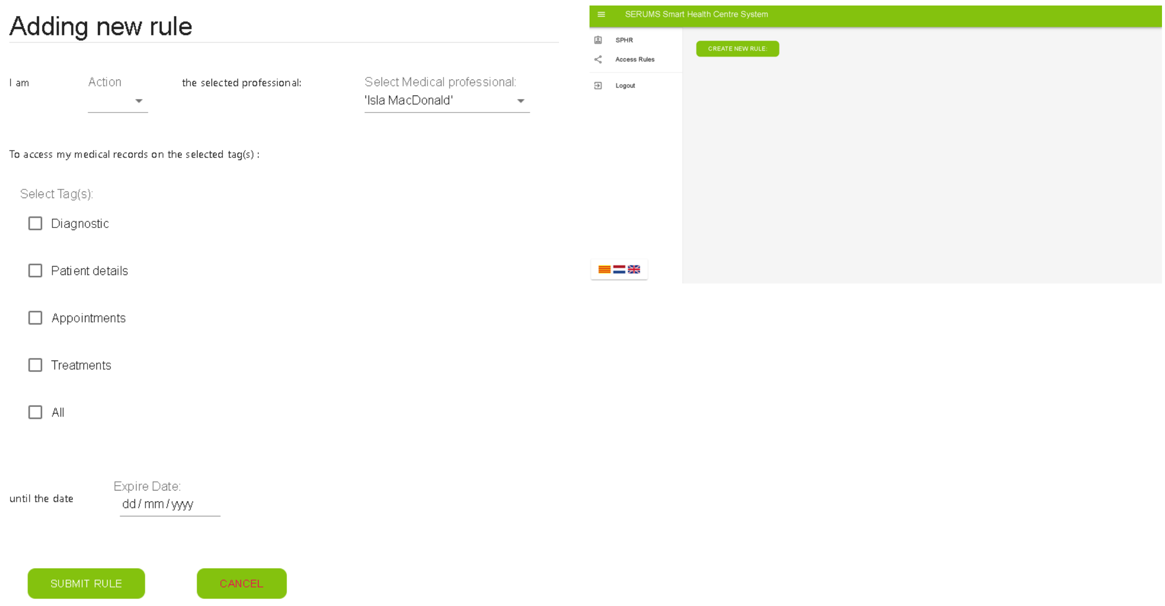
- ○ Data fetcher: it fetches the data from the Data Lake. We are using *axios library*, a JavaScript open source library to perform the HTTP request.

[4] More information about React Hooks can be found at https://reactjs.org/docs/hooks-intro.html

- Then, SHCS performs a GET HTTP request to the Data Lake via SERUMS API. By having the SCHS whitelisted in the SERUMS API, it can avoid CORS (*Cross-origin resource sharing*) issues.
- From the Data Lake, SHCS receives the following information: basic patient profile (i.e., name, nationality, height, weight), data tags (previously assigned in the Data Lake processes/WP2), registered rules, and the patient data.
- Once SHCS retrieves this information, SHCS renders the data to the following components:
    - Basic Profile component
    - Patient event component
    - Registered rule component
- In the parent component - 'Health Record' component - for the components mentioned above, SHCS parses the data from the Data Lake module and renders the information accordingly.

6.3 Blockchain module integration

The Blockchain module (WP2) is responsible for the user authorisation process and the checking of access rules before retrieving SPHR. The Access Rules Creation feature (i.e., Rule component) is another core module integrated on the SHCS. In this specific SHCS module, we develop a form for the users to create their own access rules to their SPHR (Figure 10 illustrates the form (UI) enabled in the system through the menu option "Access Rules", then clicking on the "Create new rule" button.



Adding new rule

I am Action the selected professional: Select Medical professional: 'Isla MacDonald'

To access my medical records on the selected tag(s) :

Select Tag(s):

□ Diagnostic

□ Patient details

□ Appointments

□ Treatments

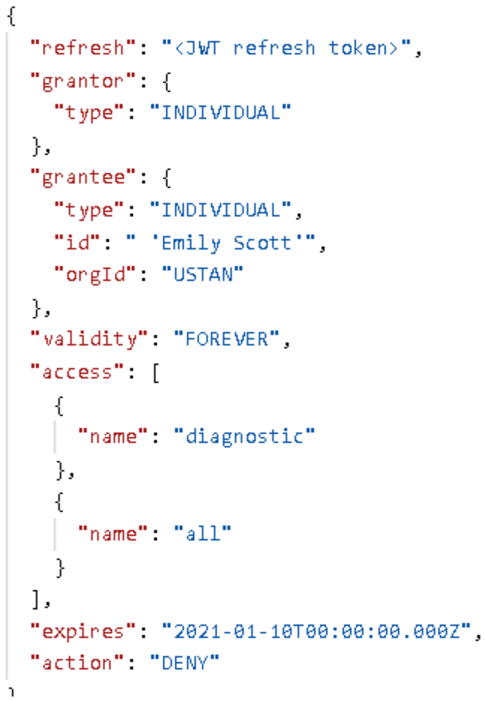
□ All

until the date Expire Date: dd / mm / yyyy

Figure 10 WUI for the Access Rules creation feature

This module concept is directly aligned with the system's high-level requirement establishing that citizens (i.e., patients) can have control of what professionals and organisations have on them in terms of data. Thus, the rules creation feature within SHCS is implemented as follows:

1. When the ‘Submit Rule’ button is clicked, SHCS performs the POST request to the Blockchain module via SERUMS API.
2. If the rule is successfully created, SHCS then performs another POST request to the Blockchain module and registers the rule there.
3. Because we need to use different tokens for creating a specific rule, SHCS provides the *Refresh token* to the body. With the Refresh token, the SERUMS API performs a POST HTTP request to the authentication module for getting a new JWT (Access tokens). The new token then is used as one of the HTTP header (i.e., authorization) for making another POST request to the Blockchain module.
4. Once the rule is created, SHCS fetches the created rules on the main page of the Rule component. In the current PoC2 system version, we have not implemented update and delete actions for already created rules. These features could be implemented in future system versions. Following an example of the POST request body (Figure 11).



```
{
  "refresh": "<JWT refresh token>",
  "grantor": {
    "type": "INDIVIDUAL"
  },
  "grantee": {
    "type": "INDIVIDUAL",
    "id": " 'Emily Scott'",
    "orgId": "USTAN"
  },
  "validity": "FOREVER",
  "access": [
    {
      "name": "diagnostic"
    },
    {
      "name": "all"
    }
  ],
  "expires": "2021-01-10T00:00:00.000Z",
  "action": "DENY"
}
```

Figure 11 POST request body code excerpt

The underlying schema for this implementation has a formal language and specification that has been developed by researchers in the WP6 team and submitted as a research paper (under review), which brings a formal approach to validate and further check the access rules created by a user. This feature will be integrated in further SHCS versions as a way to guarantee that no data leaks, or privacy breaches, could occur during the processing of stored and new rules created real-time.

Following, we present a brief explanation on the formal definition of an access rule as a tuple of information to guide SPHR retrieval.

NewRule = (granter (id,type); action; grantee (id,type,hospital); time; tags)

- **Granters** : e.g., patient (data subject); hospital (data controller).
- **Actions** : allowing; denying.
- **Grantees** (data recipients) : e.g., a doctor, a nurse from a hospital.
- **Time** (duration): rule’s start date and rule’s end date; forever; never.

- `**Tags** (data categories): e.g., consultation, treatment, diagnostic, device, medication, personal information, chemotherapy, comorbidities, hospitalisation, all (data).`

Patients are entitled to decide which part(s) of his/her own medical records will have access privileges for a given selected professional/user. The tags are predetermined by hospitals (i.e., SERUMS Use Cases/WP7) as they provide heterogenous types and amount of data concerning their patients' records. In SERUMS, the information about patients is completely synthetic and only used for the PoCs. The fabricated data is obtained by using IBM's Data Fabrication Platform (WP4) and defining rules for each of the use cases - the Data fabrication module [6], and stored as metadata in the Data Lake module (WP2) ruled by authorisation mechanisms on the Blockchain module [7].

Our proposed formal framework operates underneath the WUI enabling patients and other authorised users to securely manipulate these rules in natural language. Below some rules examples:

```
rule1 = ((p1; patient); allowing; (d1, doctor); (t1; t2); treatment);
rule2 = ((p1; patient); allowing; (d1, doctor); (t1; t2); personal information);
rule3 = ((p1; patient); denying; (d2, doctor); (t3; t4); (diagnostic, all));
```

About conflicting rules, after creation, we point out that we do not actually have to remove any rules from the set of rules to deal with conflict automatically in the future. A possible approach in the future is to associate priorities as an integer to access rules, and use an SMT solver [8] to always find the set of rules with the highest priority.

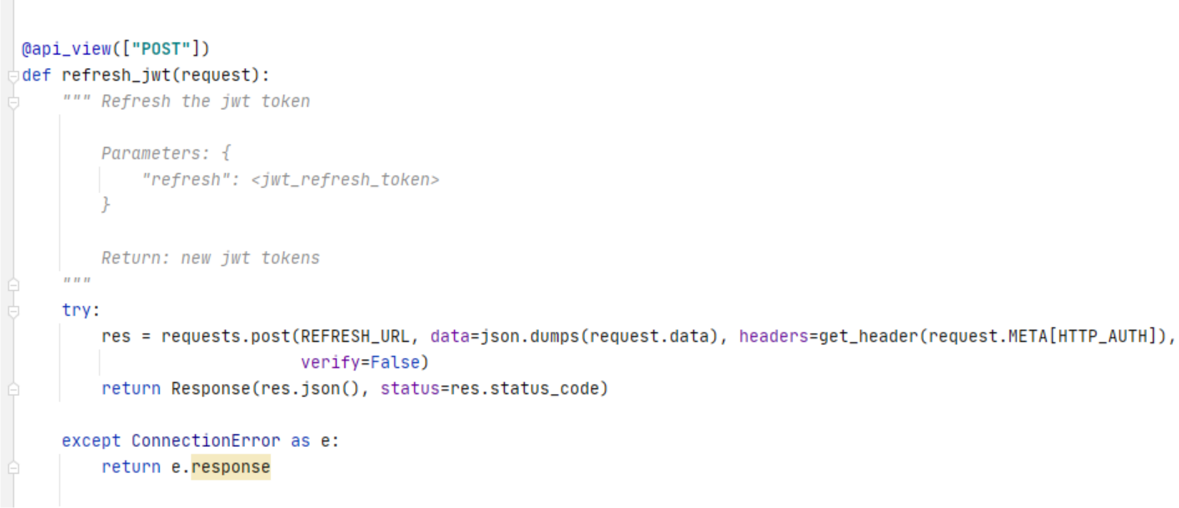
6.4 Integrated system deployment

- The SERUMS SHCS is deployed on the fracas server. It is deployed as a docker container at port 7001.
- By using SSL certificate and key, we open this client application to the public domain: https://shcs.SERUMS .cs.st-andrews.ac.uk.

SERUMS API summary:

- SERUMS API is a proxy server that connects the client SHCS and the other modules for SERUMS application. The application is written in Python using Django Rest framework[5].
- The SERUMS API contains several modules: Authentication, Blockchain, and Data Lake.
- The SERUMS API creation bypasses the CORS issues. All the modules are basically forwarding the request from the SHCS, except when the user creates an access rule. Following an example of the implemented forward function (Figure 12).
- Since we used different tokens for creating a rule, we provide the Refresh token to the body. With the Refresh token, the SERUMS API performs a POST HTTP request to the Authentication module for getting new JWT (Access tokens). The new token then is used as one of the HTTP header (i.e., authorization) for making another POST request to the Blockchain module.
- We use function instead of class to implement the proxy with support of Django Rest framework library as above mentioned.
- Each request from the SHCS requires JWT (tokens) as the Authorization header.

[5] More information about Django Rest framework can be found at https://www.django-rest-framework.org/



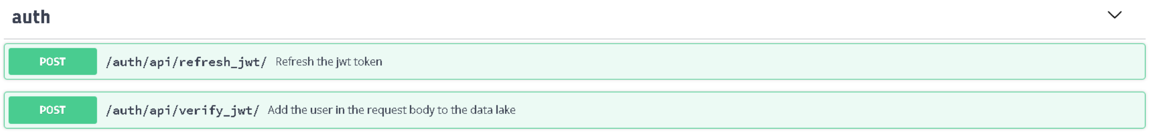
```
@api_view(["POST"])
def refresh_jwt(request):
    """ Refresh the jwt token

        Parameters: {
            "refresh": <jwt_refresh_token>
        }

        Return: new jwt tokens
    """
    try:
        res = requests.post(REFRESH_URL, data=json.dumps(request.data), headers=get_header(request.META[HTTP_AUTH]),
                            verify=False)
        return Response(res.json(), status=res.status_code)

    except ConnectionError as e:
        return e.response
```

Figure 12 SHCS forward function code excerpt

The **SERUMS API endpoints** are the following:

- Authentication module:

  We perform the HTTP request to `https://ua-web/ua`. It is the address for the authentication module container in Fracas server.



auth

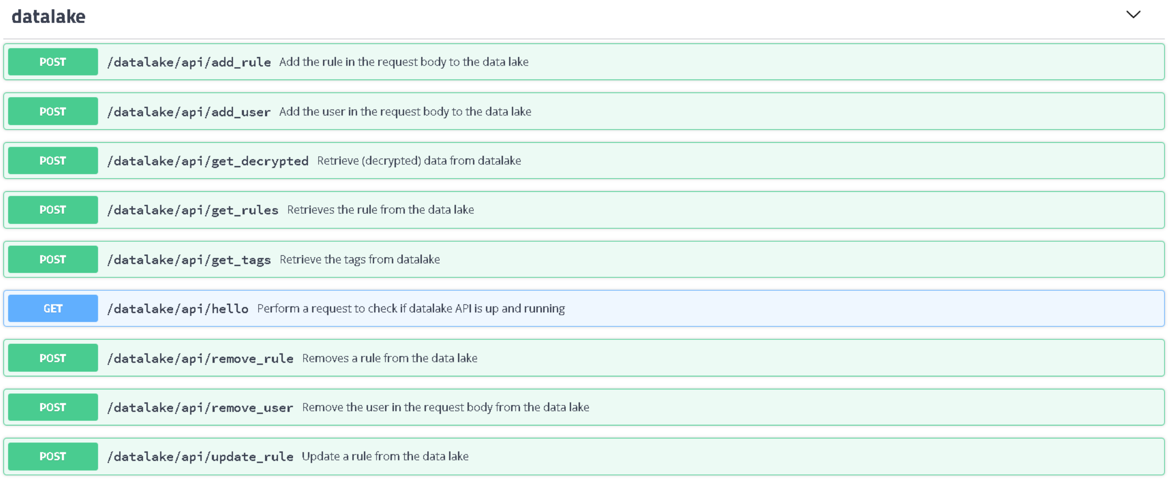
POST `/auth/api/refresh_jwt/` Refresh the jwt token

POST `/auth/api/verify_jwt/` Add the user in the request body to the data lake

Figure 13 Authentication endpoints detail

- Data Lake module:

  Address/container name in fracas: `http://data_lake_v3:5000/`



datalake

POST `/datalake/api/add_rule` Add the rule in the request body to the data lake

POST `/datalake/api/add_user` Add the user in the request body to the data lake

POST `/datalake/api/get_decrypted` Retrieve (decrypted) data from datalake

POST `/datalake/api/get_rules` Retrieves the rule from the data lake

POST `/datalake/api/get_tags` Retrieve the tags from datalake

GET `/datalake/api/hello` Perform a request to check if datalake API is up and running

POST `/datalake/api/remove_rule` Removes a rule from the data lake

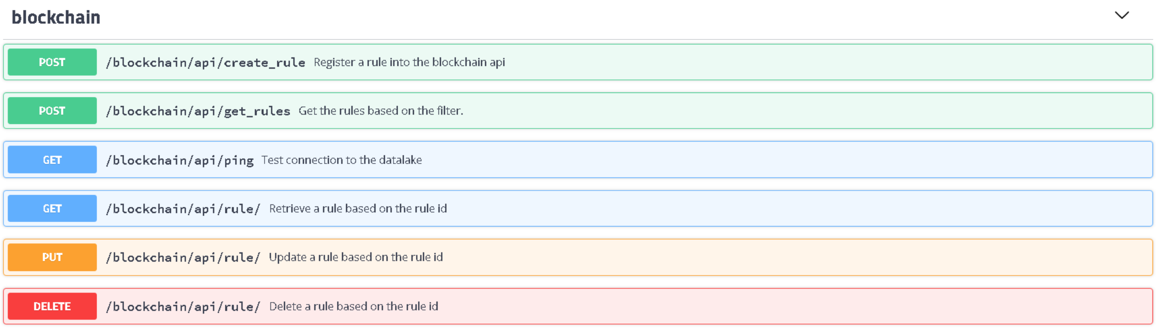
POST `/datalake/api/remove_user` Remove the user in the request body from the data lake

POST `/datalake/api/update_rule` Update a rule from the data lake

Figure 14 Data Lake endpoints detail

- Blockchain module:

  Unlike the other SERUMS partners' modules, the Blockchain module is deployed on Kubernetes cluster on `http://192.168.122.24:30001/v1`



blockchain

POST /blockchain/api/create_rule Register a rule into the blockchain api

POST /blockchain/api/get_rules Get the rules based on the filter.

GET /blockchain/api/ping Test connection to the datalake

GET /blockchain/api/rule/ Retrieve a rule based on the rule id

PUT /blockchain/api/rule/ Update a rule based on the rule id

DELETE /blockchain/api/rule/ Delete a rule based on the rule id

Figure 15 Blockchain endpoints detail

- IBM Data Fabrication module:

  Currently there is no proxy implemented between SERUMS API and IBM data fabrication tool. This will be another feature that could be implemented in future versions.

- Evaluation Questionnaire module:

In order to evaluate the integrated system as a Second Proof-of-Concept (PoC2), we also include Dutch and Catalan translations on the WUI (i.e., front-end features). We have a language component to translate the page to several different languages. We forward this component as React properties to each module. When a language is selected, the language component provides the correct translation.

6.4.1 Deployment summary

The SERUMS Proxy API is deployed in a Docker container in the Fracas server:

- ○ Container name: SERUMS api_SERUMS api_1
- ○ address: http://172.0.0.1:7002

Figure 16 presents the description of the SERUMS system integration.

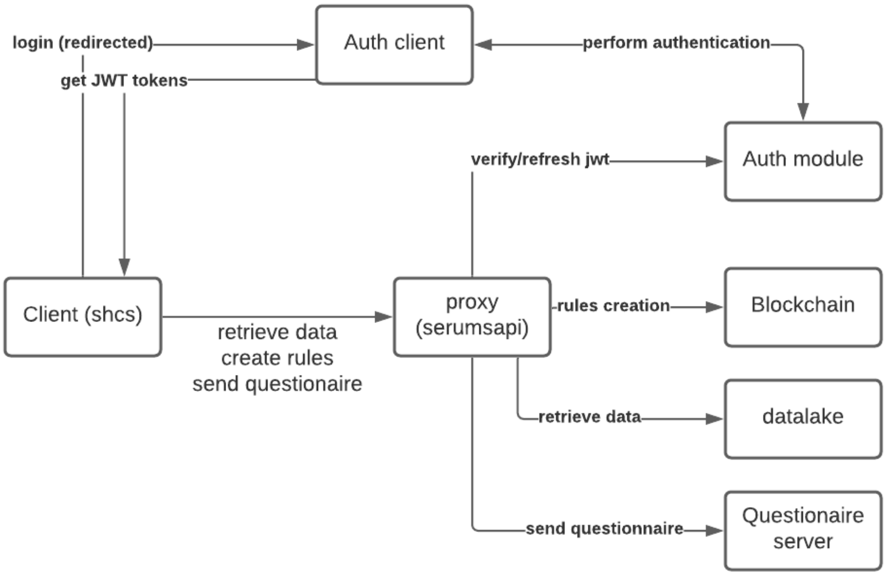


Figure 16 SERUMS system integration

Concerning Figure 16 the details are explained below:

- User access the client application via the public domain: https://shcs.serums.cs.st-andrews.ac.uk/
- Since we do not store any data in the user's local client application (no cookies), the user is redirected to the Authentication client.
- User will then register/login via the Authentication client; once login, the user will be redirected back to the SERUMS SHCS Welcome page.
- If the user chooses to visualise own medical records (SPHR), the client application will make a request to the Data Lake module via proxy server (serumsapi).
- Once the data is received, the same data is rendered in the client application. At the beginning, the user will visualise data from the default access rule that is defined by the specific UC (i.e., hospital partners).
- The user can create the access rule via the Access Rule creation page in the client application.
- The client application will create the access rule to the Blockchain module via proxy server.
- The created access rule will be listed in the SPHR page, so the user can retrieve the data for that access rule in the data log.
- Before logging out, the user will be asked to fill the Evaluation Questionnaire for PoC2.
- The questionnaire will be sent to the Questionnaire API via proxy server, and then the user will be redirected to the logout page in the Authentication client.
- All the requests made via proxy server require JWT (tokens).

6.4.2 Discussion on the SERUMS toolchain testing approach

In practice, SHCS will pass through three types of testing approaches for integration based on the V-model [9]: unit testing, integration testing, and performance testing, for both the SHCS client and the SERUMS APIs. The tests for the SHCS clients mainly focus on the rendering capability of the client application. The tests for the SERUMS API focus on the integration between SHCS and each module (i.e., Authentication, Data Lake, Blockchain, and Evaluation Questionnaire).

We will also test the capabilities of each system to handle the errors gracefully. For these, we have several initial scenarios, such as: testing valid and invalid request body made from the client application (for Access rules creation and Data retrieval process); testing the validity of data type (e.g., valid/ invalid id); testing request timeout both in the client and proxy server; testing several metadata errors and their adequate handling.

Once we perform unit and integration testing, we will perform performance/load testing. Here, we simulate many requests to the client and proxy server simultaneously. To test the client application, we are aiming to use Selenium[6] (possibly written in Java) and for the performance testing we are going to use Gatling[7] (using Scala Lang).

During Poc2, we evaluated user acceptance although we are aware that User Acceptance Testing (UAT) is one of the last stages of the software development life cycle [9]. During PoC2 we performed an UAT based end-user testing as a way to collect users' opinions about the system and to detect errors or inconsistencies for further improvements in the whole system. Comments on the PoC2 (for USTAN) in Section 6.4.3 are included as they contributed to testing and improving some of the functionality of our system as part of WP6. Further details on PoC2 will be given in the context of the respective WP7 deliverable.

6.4.3 A brief report on the PoC2 at USTAN and end-user testing

The SHCS, particularly the *Flexpass* system [4] (password creation feature) and the Evaluation Questionnaire module, both have been tested during PoC2 by a group of 25 volunteers from the area local to the Edinburgh Cancer Centre (ECC). This was conducted by the USTAN team, after approval from the University of St Andrews, School of Computer Science Ethics committee. The activities to attract user participation included the creation and distribution of an engaging public information flyer to describe the SERUMS project in an accessible fashion, including a QR code for interested participants to access further information from the project website.

Volunteers were recruited via social media in local Facebook groups across Fife, Tayside and Edinburgh to ensure only local patients were interviewed. Potential participants were asked to express their interest in the SERUMS project by email to the Serums local email address (serums-local@st-andrews.ac.uk). After that, applicants were interviewed in real time by video link on Teams. This allowed a high degree of confidence in both the participant's legitimacy and locality.

The USTAN team devised a 'COVID compliant' series of Teams-based test days, where volunteers were given access to the SHCS for the first time, supported by members of the SERUMS team in real time. Following receipt of an email registering their interest in the project, potential participants were sent a participant information pack and asked to complete a 'Doodle Poll' to indicate their availability in the proposed PoC2 schedule. Participants were scheduled across seven

[6] More information on Selenium software can be found at https://www.selenium.dev/

[7] More information on Gatling software can be found at https://gatling.io/

days, commencing on Monday 7th December, with at least two members of the USTAN team (including the system developer available online for technical support if necessary) available at each interview.

Using a pre-agreed participant's script, volunteers were asked to create their own picture password in the SHCS by choosing an image relevant to them and performed a series of tasks to test the usability of the system. This was achieved by requiring participants to share their screen with the USTAN interviewers during the *Flexpass* password creation and SHCS testing part of the process, which allowed help to be offered if required and allowed early identification of any technical issues. Participants were asked to stop screen sharing for the duration of their participant questionnaire, to preserve the confidentiality of their responses. Once they had completed the questionnaire, participants were thanked for their contribution and reminded that they would receive emails requesting them to log into the system on days 1, 3 and 6 (post-study process) using their initial access (created password) to the SHCS.

Participants were emailed three times with a pre-agreed email format to request them to login to the system on their defined days. Following two successful (or attempted) logins, each participant was emailed a link to activate their £10 Amazon Voucher in retribution for their time testing the SHCS and the use of their broadband. Both survey participants and interested volunteers who were not interviewed due to the limit on participation numbers will be emailed, thanking them for their time and asking for them to consent or not to receive further information on the SERUMS Project's progress via the quarterly newsletter. This is performed with the aim in maintaining a high level of public interest in the project and also to provide a 'pool' of potential volunteers for future rounds of system testing (PoC3).

PoC2, will give us information on the lessons learned for the future improvement of SHCS features and to reach adequate interoperability among technologies. We added one important feature to the client application: the error handling and coding refinement that has been overlooked before PoC2. At the time we conducted PoC2, the system only had errors documented in the system log. From now on, the client application also informs the user if a given error occurs. A positive outcome of performing the PoC2 before reporting D6.2 is that all SHCS modules and APIs have important feedback from a users' perspective, which is the major goal of SERUMS project.

7 SERUMS Integrated System: formal verification

In parallel to the development of SERUMS SHCS and APIs, we are building a formal model of the system in a formal language (here timed automata). This model allows us to perform formal verification of system properties. The model includes all components of the SERUMS system, their communication, and components representing additional possible user behaviour.

We use the Uppaal tool[8] that provides an expressive modelling formalism and a family of model checkers. Uppaal has been used in multiple projects to address similar challenges, for example [5, 10].

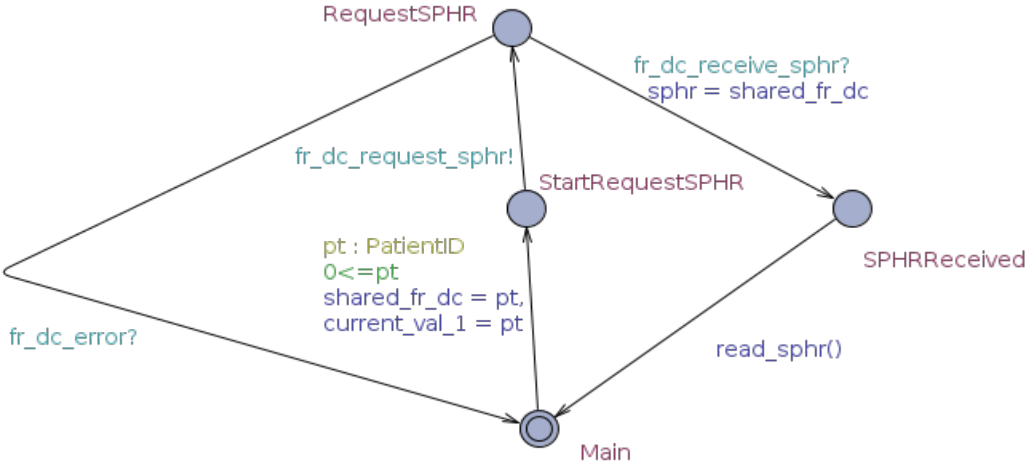


Figure 17 Example automaton

Models in Uppaal are defined as networks of timed automata. An example of an automaton is shown in Figure 17. An automaton can be considered as a graph where nodes are states of the system (circles on figures, can have cherry coloured names) and edges are transitions defining how the system changes state (black arrows between states). Transitions have a set of optional labels:

- *Local variable definitions* (brass labels) to be used in the other labels (only valid within the same transition).
- A *guard* (green label) is a Boolean expression controlling the enablement of the transition.
- A *channel* allows automata to synchronize actions (cyan labels). This label contains a special channel variable and either '!' or '?'. Two automata having transitions labelled with the same channel must take these transitions simultaneously. One of the transitions marked with '!' is an initiator of the synchronisation or a sender, another with '?' is a receiver.
- An *update* is a sequence of actions that modify the variables of the model (dark blue labels). The updates are defined with a subset of C language and can refer to functions declared separately.

Several timed automata are combined into a network via synchronizations and shared variables. Note that the same automaton (called a 'template' in Uppaal) can be instantiated multiple times in the network: all instances are independent but have identical behaviour. For example, to model multiple patients, a single template is created and multiple patient automata are instantiated. At each point of time the network has three options to evolve to the next state: 1) by passing time 2) by

[8] More information on Uppaal tool can be found at http://www.uppaal.org/

one automaton making a transition that is not synchronized with any other automata 3) by automata making a simultaneous transition synchronized over the same channel.

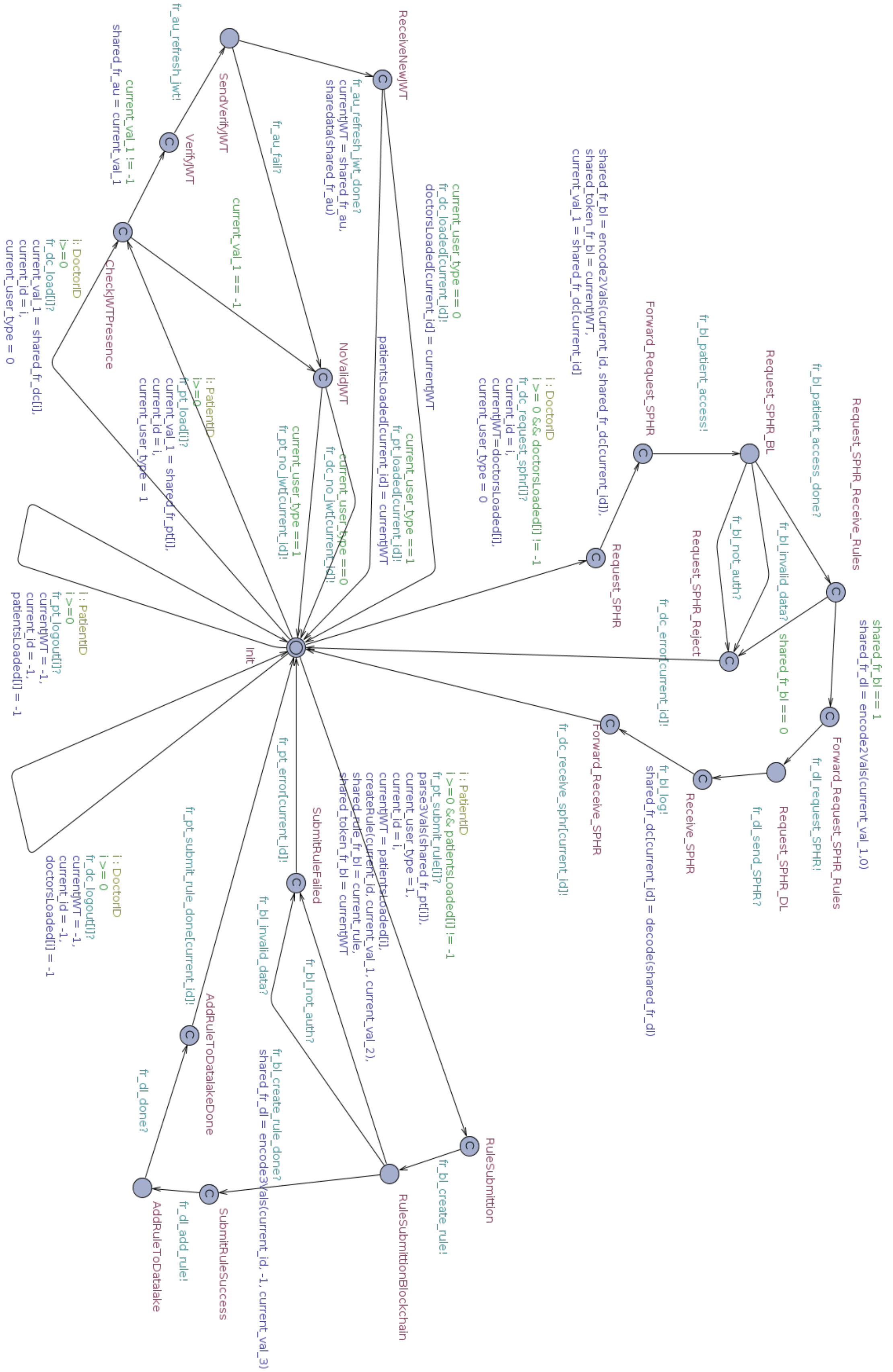


Figure 18 SHCS model

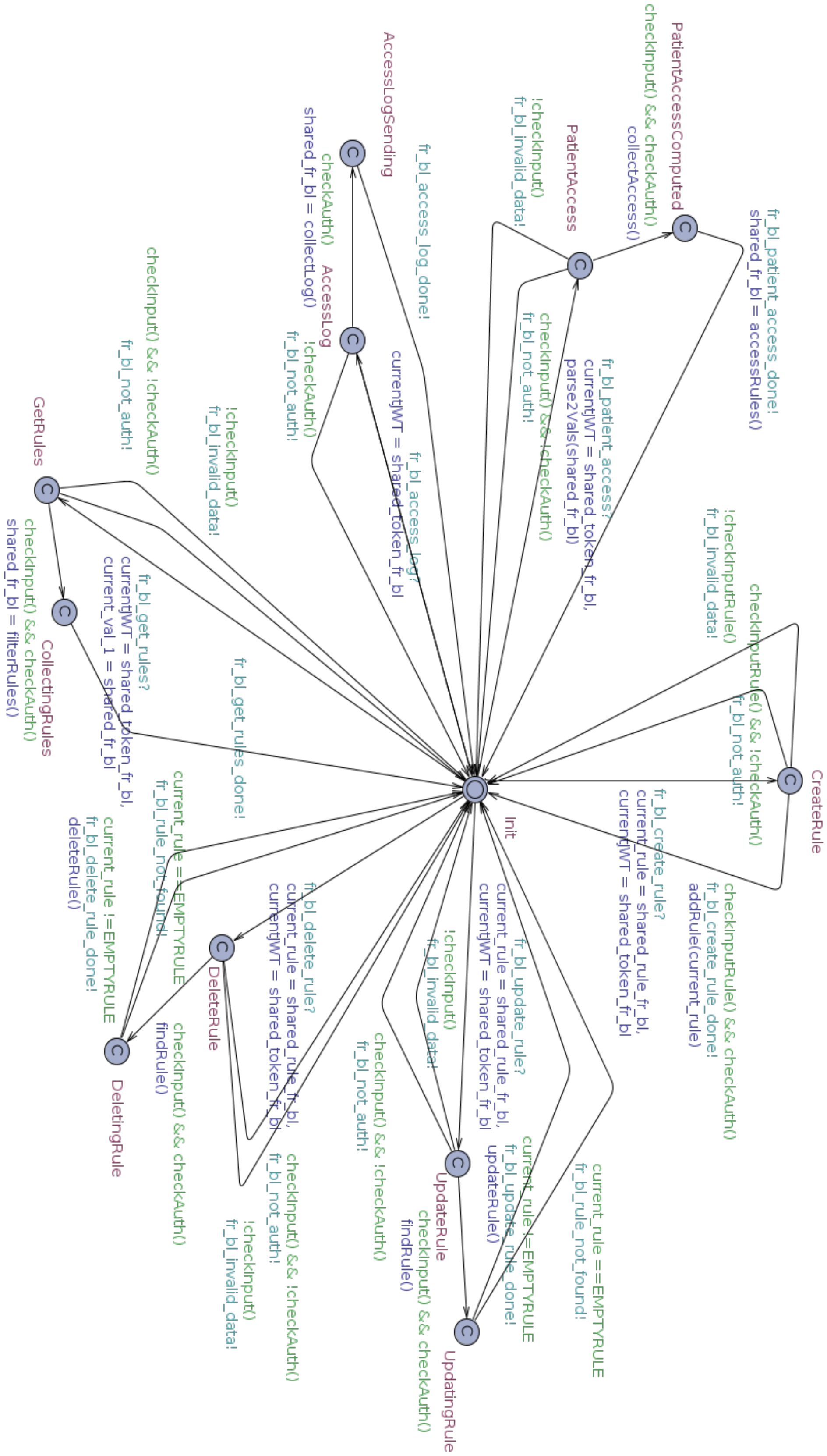


Figure 19 Blockchain model

7.1 Model of the SERUMS Integrated System

The current version of the model is based on the implementation developed for the second Proof-of-Concept (PoC2). The model consists of 8 templates: SHCS, Blockchain, 2 template automata for Data Lake, and 4 template automata for Authentication. Two additional automata describe behaviours of patients and doctors.

The SHCS automaton, modelling the SHCS module and the SERUMS API, is shown in Figure 18. The central state is the initial state of the automaton (Init). The automaton has the following parts:

- Behaviour to the left of the 'Init' state describes actions taken during the connection of a user to the SHCS. If the user has a JWT, the component asks the authentication system to check the token and logs the user in. In case of non-present or invalid JWT, the user is forwarded to the authentication system.
- Creation of rules for the access to patients' data is in the right part of the automaton. A patient is required to be logged in and to have a valid JWT. An information about doctor id, tag, and whether the rule would allow or deny access is transferred from the patient via a shared variable. At the next step the SHCS sends a request to the Blockchain and afterwards to the Data Lake automata. In the current version, the request for an additional access token (Section 6.3) is not included in the automaton.
- A request for an SPHR by a doctor is shown in the top part of the automaton. At the beginning the Blockchain is requested for the access rules. If the doctor is allowed to receive data about the patient, the corresponding request is sent to the Data Lake. Note that the request to Blockchain is not yet implemented in the Integrated System and the behaviour originates from the Information Flow Viewpoint (Appendix I).

Figure 19 illustrates the automaton for the Blockchain module. This automaton models the interactive behaviour of the Blockchain module with the rest of the Serums system. It can receive different requests from the SHCS component and (after internal validation) reply back. The requests include creation and modification of rules as well as check for patient's access rules. In the model we abstracted the notions of rules by maintaining the access matrix for doctors and patients. Creation and modification of rules modify the corresponding cells of the matrix.

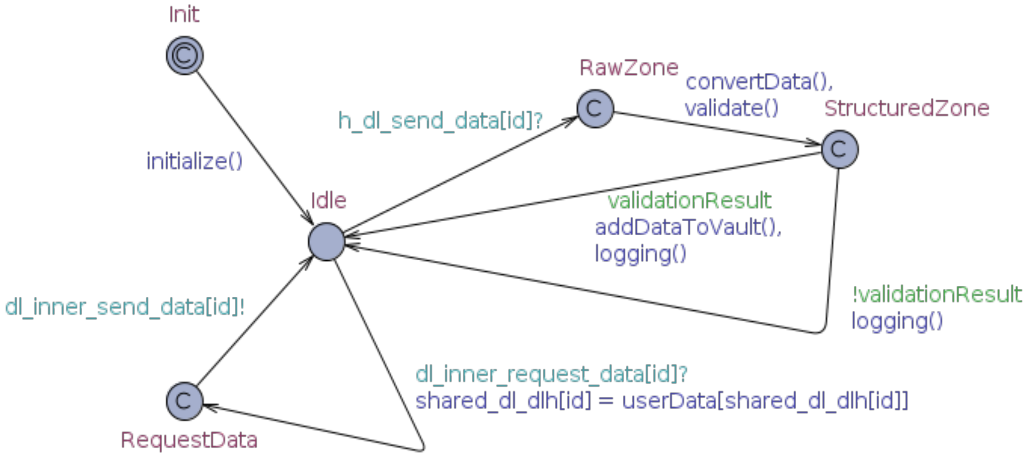


Figure 20 Model of Data Lake local instance

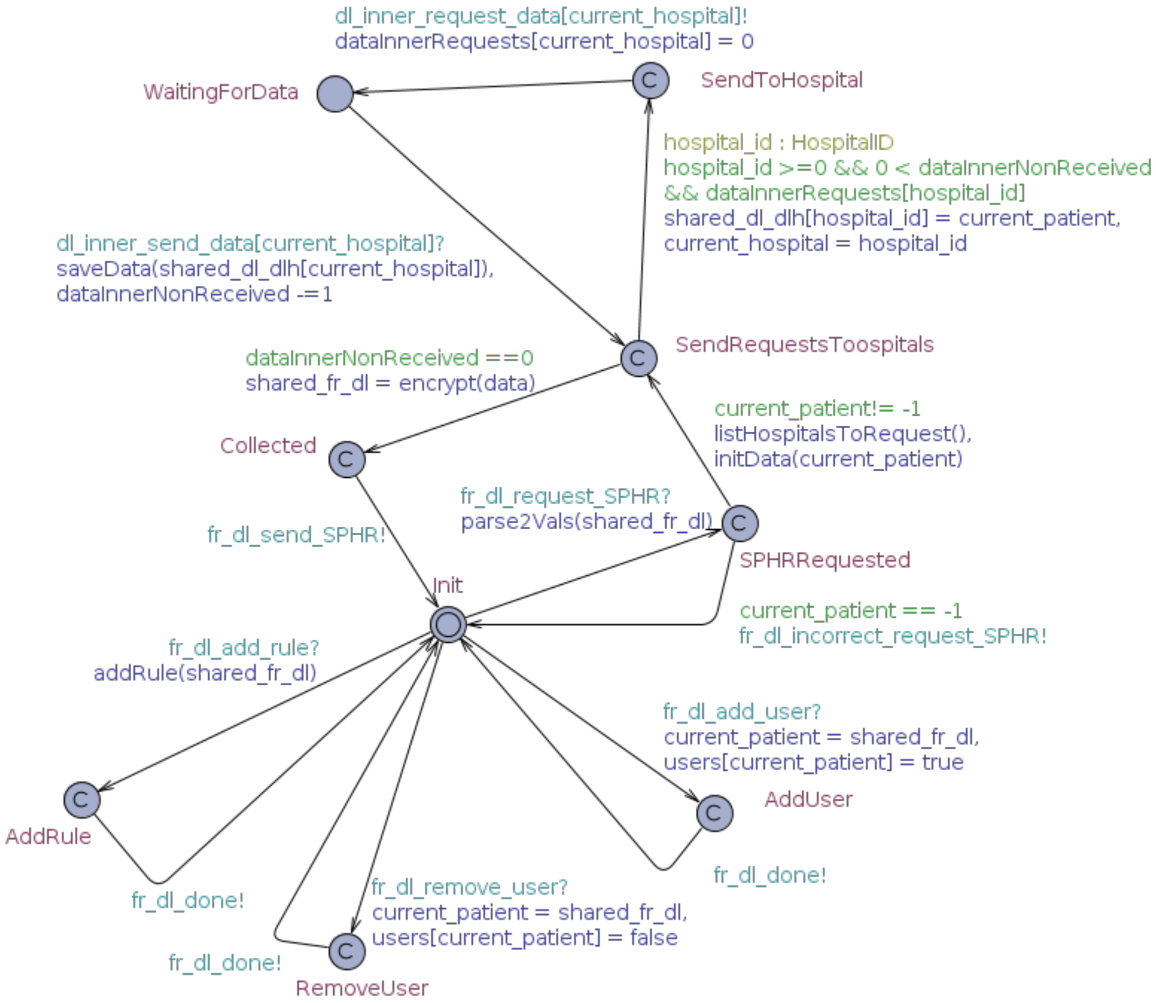

dl_inner_request_data[current_hospital]!
dataInnerRequests[current_hospital] = 0
WaitingForData
C
SendToHospital
hospital_id : HospitalID
hospital_id >=0 && 0 < dataInnerNonReceived
&& dataInnerRequests[hospital_id]
shared_dl_dlh[hospital_id] = current_patient,
current_hospital = hospital_id
dl_inner_send_data[current_hospital]?
saveData(shared_dl_dlh[current_hospital]),
dataInnerNonReceived -=1
C
SendRequestsToospitals
dataInnerNonReceived ==0
shared_fr_dl = encrypt(data)
Collected
C
current_patient!= -1
listHospitalsToRequest(),
initData(current_patient)
fr_dl_request_SPHR?
parse2Vals(shared_fr_dl)
C
fr_dl_send_SPHR!
SPHRRequested
Init
current_patient == -1
fr_dl_incorrect_request_SPHR!
fr_dl_add_rule?
addRule(shared_fr_dl)
fr_dl_add_user?
current_patient = shared_fr_dl,
users[current_patient] = true
C
AddUser
C
AddRule
fr_dl_done!
fr_dl_done!
fr_dl_remove_user?
current_patient = shared_fr_dl,
users[current_patient] = false
fr_dl_done!
C
RemoveUser

Figure 21 Model of Data Lake global level

The Data Lake module is modelled by two automata shown in Figures 20 and 21. The former one presents local instances of the Data Lake that can receive data from a related hospital and store it inside. The latter one describes the global level and interacts with the SHCS module. It manages 4 types of requests: add and remove users, add rule, and request SPHR. The first three interactions simply update the internal variables, while the last one requests local instances to collect the required data, aggregates the data, encrypts the data and sends the encrypted data to the SHCS. In the current version, tags are not used in the SPHR creation, all patient's data is sent provided the presence of a rule allowing to get the data. The model will be extended to select the data according to the tags.

Figures 22 and 23 show the models of patients and doctors. These two models share a large common part corresponding to the authentication (login and sign up). Both automata start at the bottom-right state and try to connect to SHCS. If there is a valid JWT (have signed up and logged in before) the SHCS would transfer them to the Main state, otherwise to the PGAView. From the latter state they would follow authentication procedures (see deliverable D5.3 for details). After being logged in, patients can create rules for the data access and doctors can try to request SPHRs.

Automata describing the behaviour of the Authentication module are presented in detail in the deliverable D5.3.

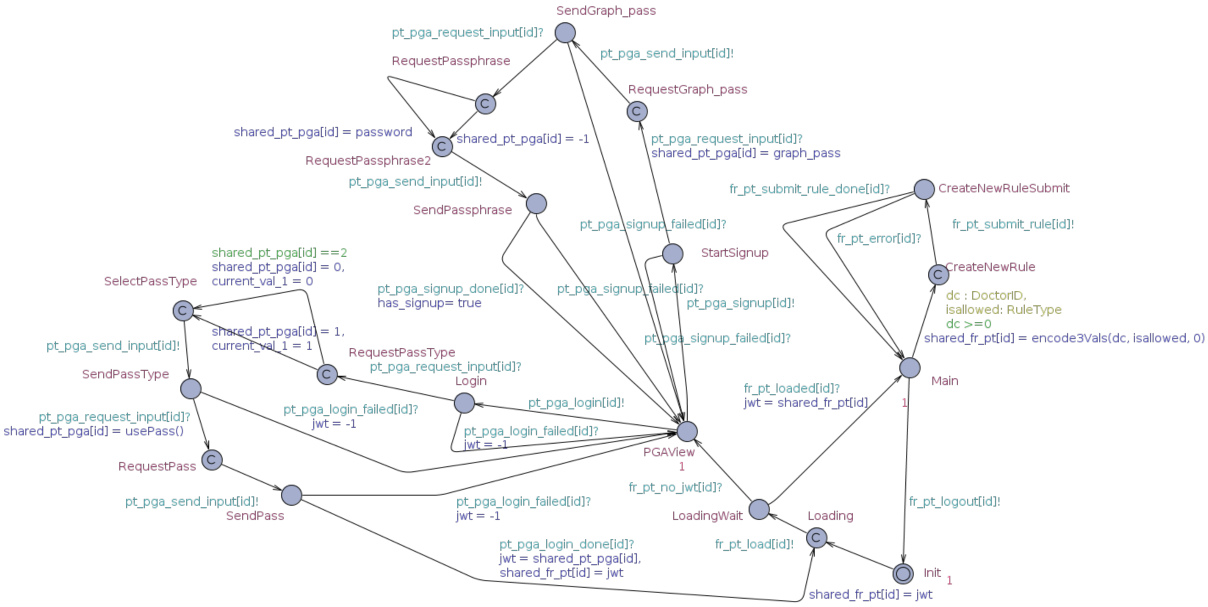


Figure 22 Model of a patient

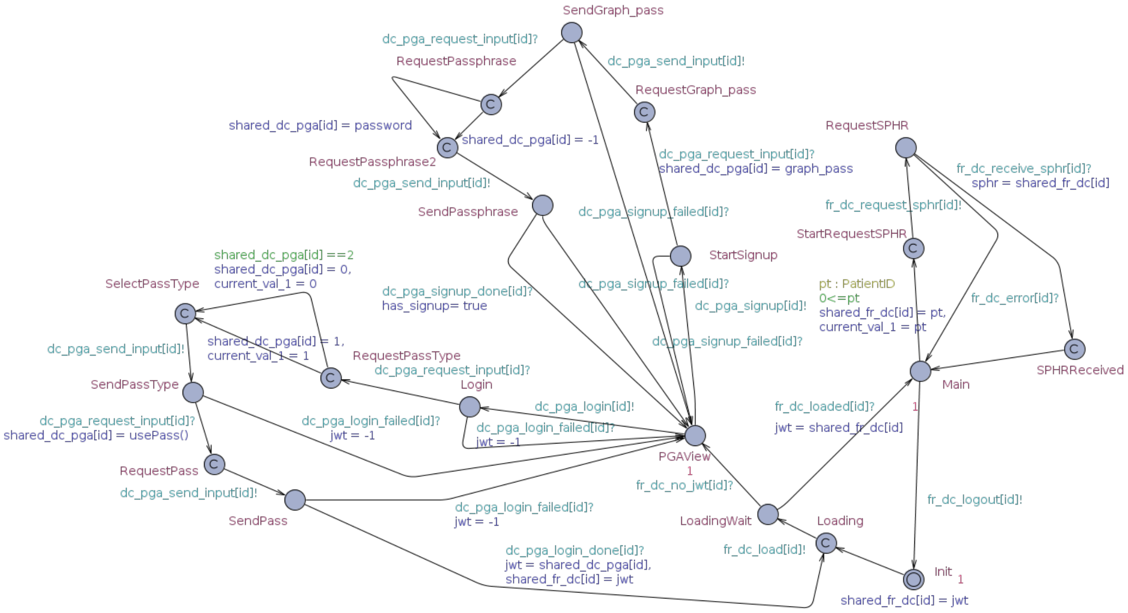


Figure 23 Model of a doctor

7.2 Property Verification

Uppaal provides two mechanisms for the verification of properties. The first mechanism is the Uppaal model checker (MC). The properties shall be expressed in the Uppaal query language based on a simplified version of Timed Computation Tree Logic (TCTL). The result of a query can be either 'property is satisfied' with a mathematical guarantee or 'property is violated' and a trace how to violate it.

Unfortunately, model checking is infeasible for large models. This problem is also known as the state-space explosion problem. To address this problem, an alternative approach based on algorithms from statistics was proposed. The core idea of Statistical Model Checking (SMC) [11–13] is to make many simulations of the model during which properties are monitored. Statistical algorithms are used to decide the probability of the property to be satisfied with some degree of confidence. Being simulation-based, SMC is known to be less time and memory consuming that model checking and, therefore capable to verify larger models.

Uppaal SMC [14] is a statistical model checker for stochastic timed automata models included in Uppaal distribution. Stochastic extension adds probabilistic choice between transitions and probability distribution for time delays. For the queries, Uppaal SMC uses an extension of Metric Interval Temporal Logic (MITL). It provides queries to check probability estimation – probability of the property to be satisfied within a given time bound.

In our work we are using Uppaal model checking to verify properties that can be checked on small isolated parts of the model. To make it isolated, we are either removing synchronisations with other parts of the models or build a small 'environment' that can perform required synchronisations. For the properties that would require the full model we are using Uppaal SMC.

At this stage we have built a model and checked it on several properties. Examples are listed in the Table 1. In the following queries we would use an expression of a form *A[] p* which means that the property *p* is required to hold in all states on all execution paths.

Table 1 Examples of properties for model checking and details

Property	Formula	Modules	Checker	Status
No deadlock, i.e. no state where automata cannot progress	*A[] !deadlock*	Authentication module, Request for SPHR part	MC	Verified
User with wrong password cannot receive a JWT	*A[] (Patient.JWT== -1)*	Authentication module	MC	Verified
	Patient is modified to input an incorrect password. Formula states that the initial value is unchanged during execution			
An issued JWT can be verified by the Authentication module	*A[] (!SHCS.NoValidJWT)*	Authentication + SHCS modules	MC	Verified
	Patients and Doctors initialized with valid JWT. Assumption: JWT has no end of validity and users do not modify it. Formula states that NoValidJWT state of SHCS automaton is not visited.			
A doctor can only receive an SPHR for a patient if the patient has allowed that by a rule	*Pr[<=10000] ([](forall d:doctor (d.SPHRReceived => (d.sphr != -1 && bl.rules[d.sphr.patient][d]))))*	Whole system	SMC	Verified with a confidence parameter 0.99
	The property states that at any moment if a doctor is in the state 'SPHRReceived' then the doctor has received a non-empty SPHR for a patient and there is a blockchain rule (valid at this moment) allowing him/her to receive the data about the patient. *Pr[<=10000]* indicates that Uppaal SMC considers traces up to 10000 steps long.			

SPHR does not contain the data not allowed by rule tags	*A[] (forall d:doctor (d.SPHRReceived => (d.sphr != -1 && dl.sphr.data allowed by dl.rules[d.sphr.patient][d])))*	Data Lake + Blockchain + SHCS	MC?	Requires further development of the model

8 Conclusion

This document is the second deliverable of Work Package 6: “Integration and Testing” focusing on the backend and frontend development of the Smart Health Centre System (SHCS) software. It shows the technologies integration to enable a user to perform the different actions in the system and further evaluate it through an online questionnaire right after logout.

This deliverable reflects work in progress and will need further input from the partners on each one of the three use cases (USTAN, FCRB and ZMC) to be as realistic and feasible as possible, and will thus be refined for PoC3 over the next couple of months.

8.1 Next steps

Deliverable D6.3 (due month 36) will describe more refined and advanced versions of the integration and testing of all the SERUMS technologies and their application for the development of the SERUMS Smart Health Centre System (SHCS).

The technical perspective of the development brings important tasks such as the implementation of several functional and non-functional tests for the SHCS and proxy client, for example, unit testing, integration testing, user acceptance testing (PoC3), and performance testing.

Furthermore, we aim the implementation and development of several other sections in modules for the SHCS be tailored for professional user profiles. This task will entail that partners from hospitals (ZMC, FCRB) provide to the USTAN team the UI requirements (e.g., design, wireframes) for the WUI development for professionals (e.g., doctors, nurses, admin). Wireframing is a way to design a website service at the structural level. A wireframe is used to lay out content and functionality on a page which takes into account the user needs.

The SHCS will also be integrated with complete Access Rules feature coding underlying the Blockchain module, including requirements such as create/retrieve/update/delete operation over rules, and few other details on parameters (duration, professional description, etc.).

The Uppaal model will be further extended following the development of the SERUMS SHCS. Safety and security properties will be formalised and checked on the model. For the security properties attackers will be modelled given different levels of power. This will exploit attack models to explore how to violate the correct behaviors of the system and find potential vulnerabilities. The properties that are going to be considered include: checking functionality access based on user type, unauthorised access to data, replay attacks, man-in-the-middle attacks, logging and log availability.

Finally, the integration of the data generated from IBM Data Fabrication to the Data Lake will allow a more complete evaluation of the system with users and new added features will compose a trustworthy healthcare platform.

References

[1] The SERUMS tool-chain: Ensuring Security and Privacy of Medical Data in Smart Patient-Centric Healthcare Systems, Janjic, V., Bowles, J. K. F., Vermeulen, A. F., Silvina, A., Belk, M., Fidas, C., Pitsillides, A., Kumar, M., Rossborry, M., Vinov, M., Given-Wilson, T., Legay, A., Blackledge, E., Arredouani, R., Stylianou, G. & Huang, W., 2019 IEEE International Conference on Big Data, 9-12 December 2019, Los Angeles, USA. DOI: 10.1109/BigData47090.2019.9005600

[2] Interacting with next-generation smart patient-centric healthcare systems, Bowles, J., Mendoza-Santana, J., Webber, T., Adaptive and Personalized Privacy and Security Workshop (APPS 2020), UMAP (Adjunct Publication), 2020. DOI: 10.1145/3386392.3399561

[3] Designing a Patient-Centric System for Secure Exchanges of Medical Data, Webber, T., Santana, J.M., Vermeulen, A.F., Bowles, J. K. F., In: Gervasi O. et al. (eds) Computational Science and Its Applications – ICCSA 2020. SEPA 2020 Workshop (online). LNCS (vol.12254), Springer, Cham, 598-614, 2020. DOI: 10.1007/978-3-030-58817-5_44.

[4] Design and Development of a Patient-centric User Authentication System, Constantinides, A., Belk, M., Fidas, C., Pitsillides, A., Adaptive and Personalized Privacy and Security Workshop (APPS 2020), UMAP (Adjunct Publication), 2020. DOI: 10.1145/3386392.3399564

[5] TAMAA: UPPAAL-based mission planning for autonomous agents, Gu, R., Enoiu, E., Seceleanu, C., In: Proceedings of the 35th Annual ACM Symposium on Applied Computing, 2020. DOI: 10.1145/3341105.3374001

[6] On defining rules for cancer data fabrication, Bowles, J., Silvina, A., Bin, E., Vinov, M., Rules and Reasoning: RuleML+RR 2020, LNCS, Springer, 2020. DOI: 10.1007/978-3-030-57977-7_13

[7] Integrating Healthcare Data for Enhanced Citizen-Centred Care and Analytics, Bowles J., Mendoza-Santana, J., Vermeulen, A. F., Webber T., Blackledge E., EFMI STC 2020, Integrated Citizen centered digital health and social care Citizens as data producers and service co-creators, Virtual conference, 26-27 November 2020, DOI: 10.3233/SHTI200686

[8] Z3: An Efficient SMT Solver, de Moura L., Bjørner N., In: Ramakrishnan C.R., Rehof J. (eds) Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2008, LNCS (vol. 4963), Springer, Berlin, Heidelberg, 2008. DOI: 10.1007/978-3-540-78800-3_24

[9] Advancements in the V-model, Mathur, S., Malik, S., International Journal of Computer Applications, vol 1(12), p. 29–34, 2010. DOI: 10.5120/266-425

[10] Statistical Model Checking of an Energy-Saving Cyber-Physical System in the Railway Domain, Basile, D., Giandomenico, F.D., Gnesi,S., In: Proceedings of the Symposium on Applied Computing, 2017. DOI: 10.1145/3019612.3019824

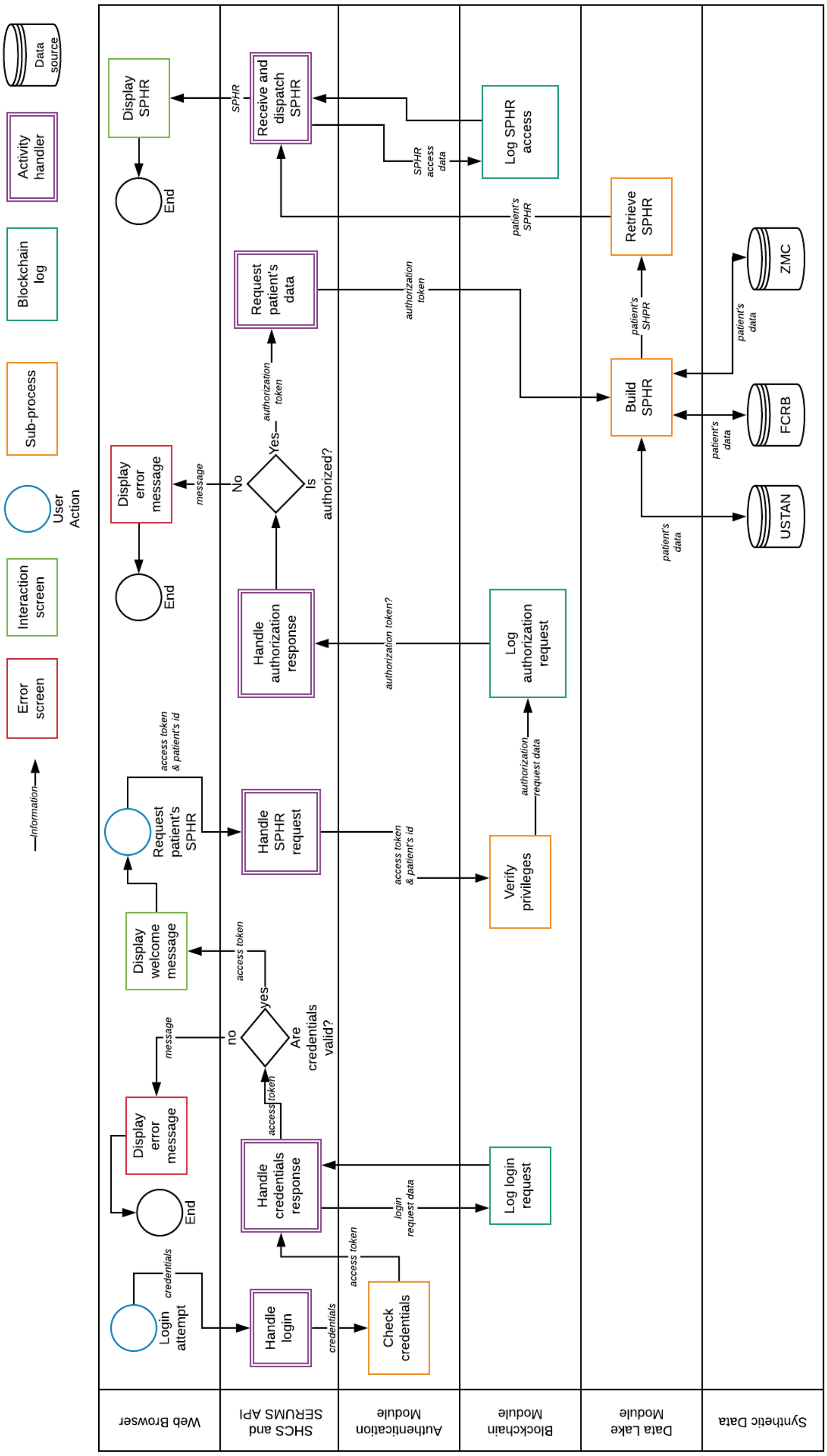
[11] Approximate probabilistic model checking, Hérault, T., Lassaigne, R., Magniette, F., Peyronnet, S., In: Proceedings of the 5th International Conference on Verification, Model Checking, and Abstract Implementations, LNCS,. Springer, vol. 2937, pp. 73–84, 2004. DOI: 10.1007/978-3-540-24622-0_8

[12] On statistical model checking of stochastic systems, Sen, K., Viswanathan, M., Agha, G., In: 17th International Conference on Computer Aided Verification, LNCS, Springer, vol. 3576, pp. 266–280, 2005. DOI: 10.1007/11513988_26

[13] Statistical model checking: An overview, Legay, A., Delahaye, B., Bensalem, S. In: International Conference on Runtime Verification, pp. 122–135, Springer, 2010. DOI: 10.1007/978-3-642-16612-9_11

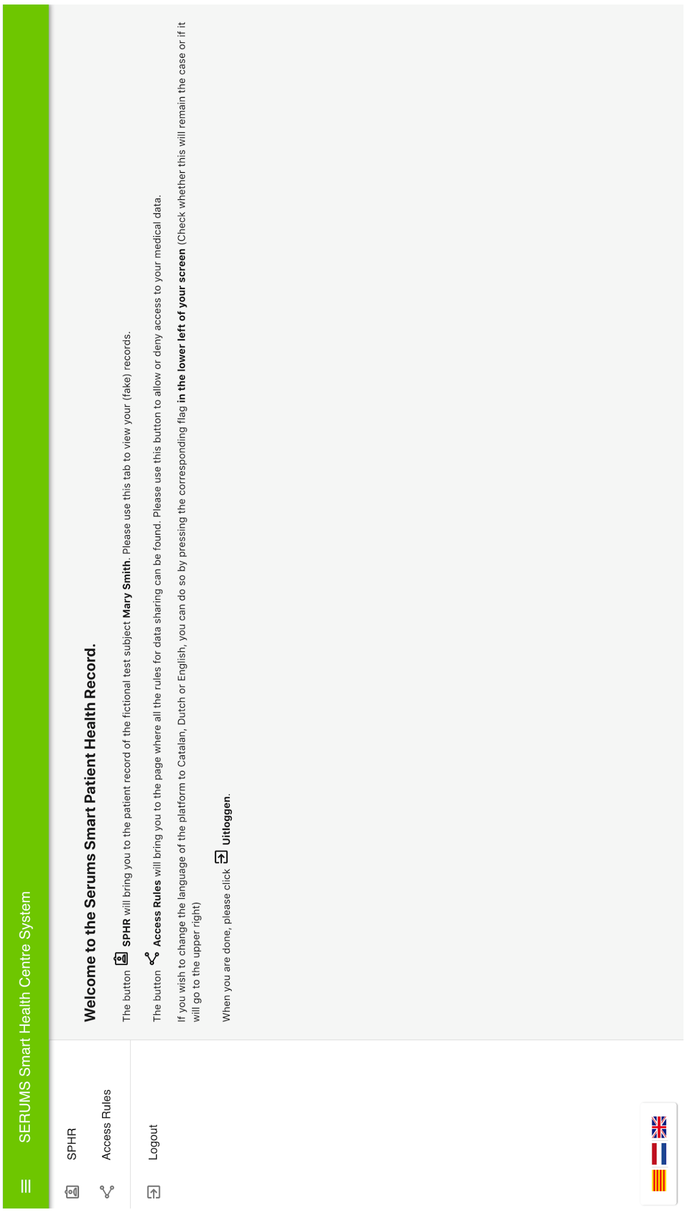
[14] Uppaal SMC tutorial, David, A., Larsen, K.G., Legay, A., Mikučionis, M., Poulsen, D.B., In: International Journal on Software Tools for Technology Transfer 17(4), pp. 397–415, 2015. DOI: 10.1007/s10009-014-0361-y

APPENDIX I - SERUMS SHCS Information Flow Viewpoint



APPENDIX II - SERUMS SHCS Web User Interface (WUI)

A. WUI - Welcome page screen



Welcome to the Serums Smart Patient Health Record.

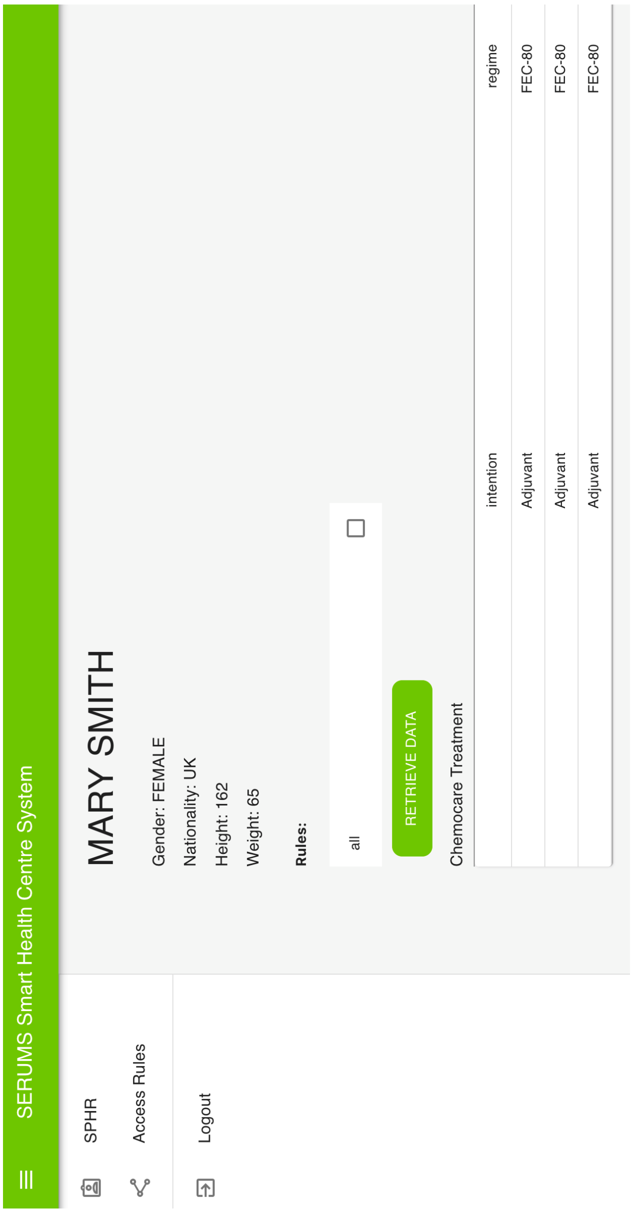
The button **SPHR** will bring you to the patient record of the fictional test subject **Mary Smith**. Please use this tab to view your (fake) records.

The button **Access Rules** will bring you to the page where all the rules for data sharing can be found. Please use this button to allow or deny access to your medical data.

If you wish to change the language of the platform to Catalan, Dutch or English, you can do so by pressing the corresponding flag **in the lower left of your screen** (Check whether this will remain the case or if it will go to the upper right)

When you are done, please click **Uitloggen**.

B. WUI - SPHR feature screen



MARY SMITH

Gender: FEMALE

Nationality: UK

Height: 162

Weight: 65

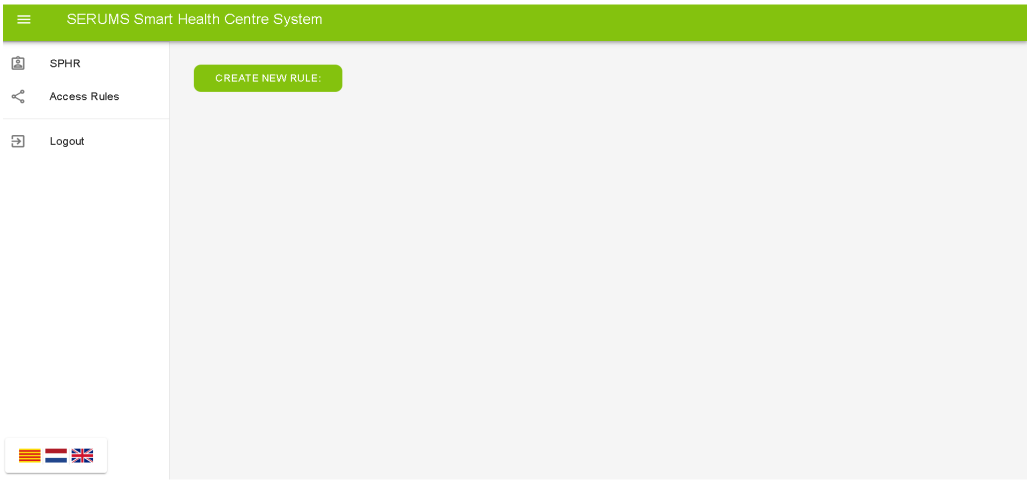
Rules:

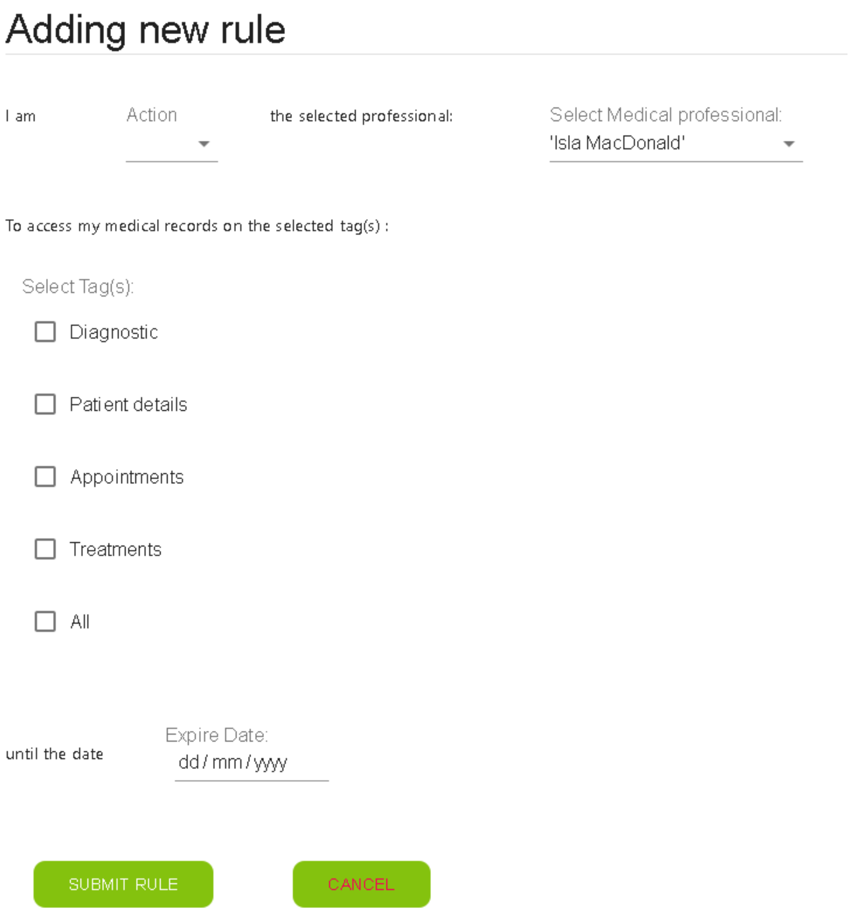
all ☐

Chemocare Treatment

intention	regime
Adjuvant	FEC-80
Adjuvant	FEC-80
Adjuvant	FEC-80

C. WUI - Rules creation screens





Adding new rule

I am Action the selected professional: Select Medical professional: 'Isla MacDonald'

To access my medical records on the selected tag(s) :

Select Tag(s):

□ Diagnostic

□ Patient details

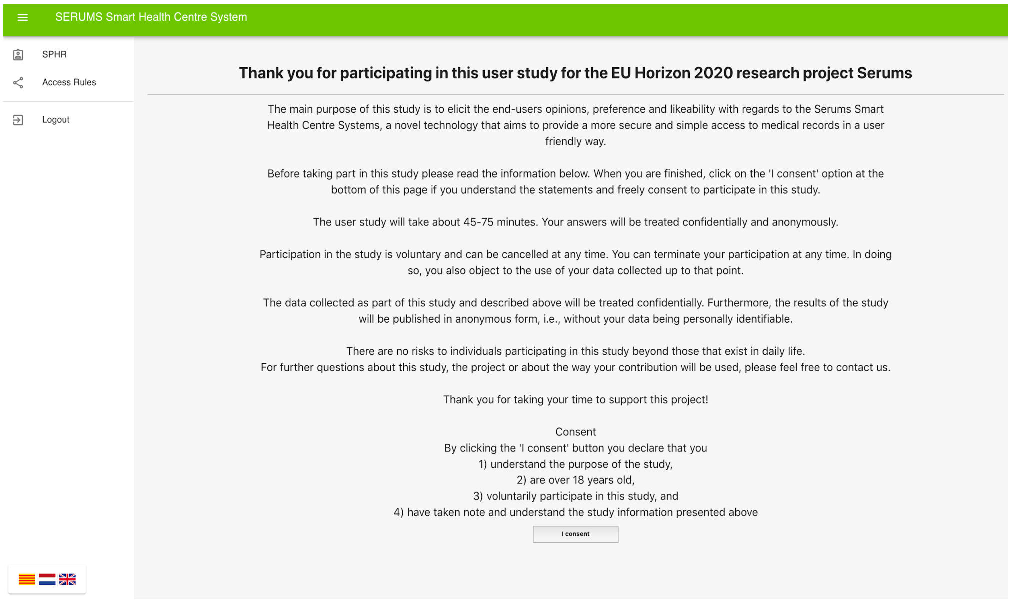
□ Appointments

□ Treatments

□ All

until the date Expire Date: dd / mm / yyyy

D. WUI - Evaluation Questionnaire screen



Thank you for participating in this user study for the EU Horizon 2020 research project Serums

The main purpose of this study is to elicit the end-users opinions, preference and likeability with regards to the Serums Smart Health Centre Systems, a novel technology that aims to provide a more secure and simple access to medical records in a user friendly way.

Before taking part in this study please read the information below. When you are finished, click on the 'I consent' option at the bottom of this page if you understand the statements and freely consent to participate in this study.

The user study will take about 45-75 minutes. Your answers will be treated confidentially and anonymously.

Participation in the study is voluntary and can be cancelled at any time. You can terminate your participation at any time. In doing so, you also object to the use of your data collected up to that point.

The data collected as part of this study and described above will be treated confidentially. Furthermore, the results of the study will be published in anonymous form, i.e., without your data being personally identifiable.

There are no risks to individuals participating in this study beyond those that exist in daily life.
For further questions about this study, the project or about the way your contribution will be used, please feel free to contact us.

Thank you for taking your time to support this project!

Consent
By clicking the 'I consent' button you declare that you
1) understand the purpose of the study,
2) are over 18 years old,
3) voluntarily participate in this study, and
4) have taken note and understand the study information presented above

APPENDIX III - Software libraries included

A. SERUMS SHCS Libraries versions

```
@date-io/core= ^1.3.6
@date-io/date-fns= ^1.3.13
@date-io/moment= ^1.3.13
@material-ui/core= ^4.9.10
@material-ui/icons= ^4.9.1
@material-ui/lab= ^4.0.0-alpha.54
@material-ui/pickers= ^3.2.10
@testing-library/jest-dom= ^4.2.4
@testing-library/react= ^9.3.2
@testing-library/user-event= ^7.1.2
axios= ^0.19.2
date-fns= ^2.11.1
fernet= ^0.3.1
js-cookie= ^2.2.1
moment= ^2.25.3
node-forge= ^0.9.1
pretty-ms= ^7.0.0
query-string= ^6.13.5
react= ^16.13.0
react-cookie= ^4.0.3
react-dom= ^16.13.0
react-material-ui-form-validator= ^2.0.10
react-router-dom= ^5.2.0
react-scripts= 3.4.0
use-global-hook= ^0.2.1
web-vitals= ^0.2.4
```

B. SERUMS API Libraries versions

```
Django>=2.0,<3.0
psycopg2>=2.7,<3.0
djangorestframework>=3.11.0
django-rest-swagger
django-cors-headers
django-rest-enumfield
```

APPENDIX IV - Brief review on data protection regulatory frameworks

A. The EU General Data Protection Regulation (GDPR[9])

GDPR defines strict regulations on the ownership and handling of personal data, creating new rights for individuals. The full GDPR is described in detail across 99 articles covering all of the technical and administrative principles around how organisations should process personal data. It establishes that citizens can have control of what information people or organisations have on them in terms of data; their rights of privacy towards shared data. Also, organisations need to demonstrate compliance with GDPR, i.e. demonstrate that they have control of how personal data is used, manipulated, stored and maintained. GDPR protects individuals with regard to the processing of personal data, and relating to the free movement of personal data.

The regulation refers to shared "personal data" (subject's data) as the information that can be used to identify a person (e.g. name, phone number, address, etc.), or any other data that combined can be used to identify a person. It also defines a "Data Controller", the one who decides how to store and process a subject's data, and a "Data Processor" who processes such data on the controller's behalf. A "Joint Controller" acts together with one or more organisations, jointly determining 'why' and 'how' personal data should be processed.

Controllers and processors are answerable to public supervisory authorities in demonstrating compliance to the rights of a subject and obligations to data protection. Joint controllers must enter into an arrangement setting out their respective responsibilities for complying with the GDPR rules. For instance, in the UK, The Information Commissioner's Office (ICO)[10] is an independent body intended to uphold information rights in the public interest with regards to individual data privacy. The *ICO Guide to Data Protection*[11] summarises the rights of individuals, and organisations' rights and obligations in this matter, with checklists included to ensure and demonstrate compliance. Due to the nature of the project, which involves both data control and specific data processing activities, SERUMS can be considered a joint controller, according to ICO's checklist.

B. Rights of the data subject within SERUMS platform

According to ICO's guide, GDPR establishes the basic rights of the data subject, and those related to SERUMS platform are: *Right to Consent, Right to Withdrawal, Right to be Informed, Right to Restriction of Processing, Right to Object, Right of Access by the Data Subject, Right to Portability*. Other established rights on GDPR are: *Right to be Forgotten (Right to Erasure), Right to Rectification, and rights in relation to automated decision making and profiling*.

Some notes concerning SERUMS compliance:

- **About the Right to Consent and the Right to Withdrawal**; Within SERUMS data subjects (patients) have the right to allow or deny access to their medical records, using a user-friendly interface to define access rules. The functionality enables patients to define the grantees, the specific medical information, and the duration for the access permission (or blocking).

[9] GDPR: [gdpr-info.eu]

[10] ICO: [https://ico.org.uk/]

[11] Guide to data protection: [https://ico.org.uk/for-organisations/guide-to-data-protection/]

- **About the Right to be Informed**; Data subjects have the right to ask for explanations of how their data have been processed, which needs to be provided by controllers/processors using clear and plain language, following the “Lawfulness, Fairness, and Transparency” data processing principles.
- **About the Right to Restriction of Processing and Right to Object**; Data subjects have the right to restrict or object to the processing of their personal data at any time, although this may only relate to a particular purpose in which the data is being processed for at a given point in time. This also allows patients to decide whether or not the information is to be retrieved to a given professional(s) and for how long.
- **About the Right of Access by the Data Subject**, SERUMS allows a patient to visualise his own medical records whenever the hospitals made information available to the platform through the SERUMS APIs.
- **About the Right to Portability**; The SERUMS platform allows patients from participating countries to enable professionals in other locations (within EU) to access their medical record and determine specifically the information that is made accessible and for how long the access is granted/denied. At this point, the platform facilitates a switch of UI language, according to users’ profile, however, the medical records remain displayed in their original language.
- **Right to be Forgotten (Right to Erasure) and Right to Rectification**; this is applicable directly to data controllers (organisations) concerning the patients’ right to exclude/erase medical and personal records. Hospitals, i.e., healthcare providers, only allow SERUMS to retrieve the data they have prepared to be shared in the platform through data tags. Regarding the Blockchain module, because its nature of immutability, we do not store personal identifiable information. Only the SERUMS ID appears on the chain and the permissions associated to the SERUMS ID; the association between patient and SERUMS ID are made off chain and thus can be deleted.
- **Concerning the rights in relation to automated decision making and profiling**; Currently, SERUMS does not apply further analysis to retrieved medical data. Moreover, SERUMS uses only fabricated data to demonstrate the usability of the platform and the purpose of the application. In order to enable the data fabrication process and refinement, patient specific data is acquired from the hospitals (anonymised), using secure transfer (encrypted, and following several security protocols).
- The current version of Serum's platform is especially about enabling functionalities complying with the **Right to Consent, Right to Withdrawal, Right of Access By the Data Subject and the Right to Portability.**

C. Data processing principles within SERUMS platform

According to ICO’s guide, the GDPR presents seven core data processing principles to guarantee that personal data is protected: *Lawfulness, Fairness, and Transparency; Purpose Limitation; Data Minimisation; Accuracy; Storage Limitation; Integrity and Confidentiality (security) and Accountability.* Thus, prioritarily, organisations must implement appropriate measures to protect sensitive data (integrity and confidentiality). The obligation applies to the amount of personal data collected, the extent of their processing, the period of their storage (in the case of SERUMS , the access privileges and their duration) and their accessibility (in the case of SERUMS , the access rules application at runtime and, consequently, the data retrieval).

SERUMS is a toolchain (i.e., a joint controller) conceived to demonstrate that such a web-based platform can enable controllers (hospitals, healthcare organisations) to share their stored medical data about data subjects (patients) to authorised recipients (healthcare professionals, or the patient himself), guaranteeing that patient rights (within governmental laws) are fulfilled.

SERUMS should guarantee that the distributed confidential data is securely integrated in the web-based platform, where the health data can be retrieved and visualised in a user-friendly front-end manner by the patient himself, and by healthcare professionals. The rightful access follows the patients' consent, meaning that SERUMS will provide a feature for access rules definition by the patients themselves, which can be seen as an explicit consent to access personal medical data.

In the context of SERUMS, GDPR supersedes organisational and local (country) regulations. Thus, patient individual rules should represent strict rules about who is allowed to access (parts of) the patient data and when. The access rules are provided by the patient including the fields that represent an explicit consent, as follows: an action (allow/deny), a grantee (professional), a selected set of medical information (data categories), and a duration.

Furthermore, SERUMS aims to enable the 'cross-border processing', that according to GDPR occurs when the processing affects data subjects (patients) in more than one Member State. In this sense, the SERUMS project includes medical partners in the EU (Use Cases - UC) with heterogeneous medical data sources governed by different access rules within respective hospitals. Thus, there is a need for examining specific access rules concerning each country and its organisations policies.

Currently, SERUMS project has three UCs in different EU countries: Scotland/UK (e.g., patients from Edinburgh Cancer Centre - ECC), The Netherlands (Zuyderland Medisch Centrum-ZMC), and Catalonia/Spain (Hospital Clinic de Barcelona - FCRB). For the purposes of the SERUMS project, **it is assumed that UCs operate in full compliance with GDPR and its legislation when storing and maintaining patients' records**. The SERUMS platform is able to retrieve patient records under this assumption **and will continue following these principles subsequent to the retrieval of the data**. It is anticipated that **UCs should only transfer information that patients themselves have the right to access and that the platform allows only authorised professionals access to it**.

Given the UCs are based on local practices in different countries, they fall under different local confidentiality and privacy-preserving regulations to which they need to conform, as follows:

- In the UK, the access of patient records within ECC, thus within the National Health Service (NHS), follows the guidance of the national GDPR working group and Information Governance Alliance (IGA). The Data Protection Act (DPA) 1998 is a piece of UK legislation that was designed to protect the privacy of personal data. GDPR reinforces it, being enacted into UK Law as the Data Protection Act (DPA) 2018. It complements the NHS Data Security and Protection Toolkit (DSPT) (NHS Official Guidance documents: Confidentiality policy and Information sharing policy) and requirements for Caldicott Guardians (i.e. senior person within a health or social care organisation who makes sure that the personal information about those who use its services is used legally, ethically and appropriately, and that confidentiality is maintained).
- In The Netherlands, ZMC is also required to comply with the MedMij standard for data and UAVG (Dutch translation of GDPR), according to the ZMC (SERUMS Dutch team).

- In Catalonia, FCRB is also required to comply with the “La Ley del Paciente” (“Lei de Autonomía del Paciente”), LOPD y futura transposición, and RGPD (Spanish translation of GDPR). The Spanish legislation (LOPD 2019) includes a local translation of the GDPR, according to the FCRB team. In addition, we also analysed the “Guía para pacientes y usuarios de la sanidad” from AEPD - the “Agencia Española Protección Datos - aepd (Noviembre 2019)”, document version: November, 2019.

D. Compliance aspects for each of the SERUMS Use Cases

GDPR Aspects	**UC1: UK**	**UC2: The Netherlands**	**UC3: Catalonia**
Patient CONSENT to the data collection, storage, processing and sharing.	The principle of **‘implied consent’** operates in the process of patient referrals (i.e. from a GP to a specialist within a local hospital). **This assumes the patient consent to the sharing of their information at the time the referral is made and for any subsequent treatment relating to the referral.** Health and social care professionals are able and even obliged to share information in the best interests of their patients within the framework set out by the Caldicott principles. Patient needs to be informed in case of data sharing; be aware of the information privacy notices (there is an ICO’s checklist to guide this).	Consent is assumed right after referral, and is considered **‘implicit consent’** since the patient agrees with the referral, consequently agrees with the information exchange. Patient data is **directly exchanged between professionals** within the hospital, caregivers, GP’s and caretakers (including a family member of a dementia patient) **without additional permission from the patient.** All ZMC professionals directly involved **may only exchange information which is necessary for the functioning of the caregivers in the treatment or supervision of the patient.**	**Consent is implicit when a patient consults a doctor for medical care.** It is not necessary for the doctor (or the health center) to request the patient consent for the collection and use of personal and health data if they are to be used for the lawful purposes. **Explicit patient consent is also not necessary** if the data processing is carried out for reasons of public interest. Health data **can also be processed without requesting consent** when the treatment is necessary to protect vital interests of the patient. Patients have the right to withdraw consent at any time, without affecting the legality of the treatment based on consent prior to withdrawal.
Patient INFORMED of data processing and sharing.	According to GDPR, patients have the right to be informed of: who is responsible for the treatment; purposes of the processing of data; legitimation for the treatment; the possibility of exercising your rights, also when transferring personal data to a third country or international organization the patient must be informed of the purposes and details, etc.		
Patient can OBJECT to the processing, or DENY access to individuals (or healthcare organisations)	**Patients may refuse for their personal data to be shared or processed**, but this request needs strong justification and will only be granted under limited circumstances.	**Patients can make a formal objection** or a formal partial objection so that some data is not shared. Also a patient can object to the data processing and exchange.	**Patients can object** to the data processing and exchange following GDPR principles.
WHO/WHAT/WHEN can have access	**Patients: they can request access to their own records.** Remark that parts can be redacted, like any third party information or anything that it is believed may cause serious harm to the patient. **Local professionals (NHS): only those who need access to personal confidential**	**Patients: they can request a copy of their own medical record.** **Local professionals (caregivers):**	**Patients: they can request a copy of their own medical record.** Organisations must provide it within a month, although the term can be extended.

	data should have access to it. The most important point here is the **purpose.** Thus, any medical staff must access patients’ data with a legitimate reason - i.e. one directly relating to the care being provided by that healthcare professional at that point in time. **External professionals:** All transfer of personal confidential data should be clearly defined, scrutinized and documented, with continuing uses regularly reviewed by an appropriate guardian.	the caregivers **have a right to access all medical information on the patient**, unless the patient makes a formal objection to this (a formal partial objection so that some data is not shared). *Note that this formal notice must be well substantiated and can be denied, therefore this shouldn’t be something that can be easily done.*	Patients can also request the medical history of his/her deceased relatives, unless the patients have objected while alive. Where a medical professional objects to a patient accessing subjective comments and amendments written about them, some portions of the medical record may be redacted. **Local professionals (Spanish law):** Access to a patient's medical record is strictly limited and will only be granted to any health professional for the specific purposes necessary to ensure efficient patient treatment. Right of access does not include the identification of health professionals who access the medical record. Doctors or health personnel can only access patients' data when there is a justification for it: medical consultation, management of health or social services, medical appointments, reasons of public interest in the field of public health. Need lawful purpose to do so. *It is assumed that when patients seek treatment, considering the hierarchical structure that holds the medical data, he agrees with the storage and sharing, all professionals related to the case and part of the hierarchical structure, will have access to the patient information. Unless the patient explicitly denies access to a person or organisation within the hierarchy, which can be improbable but not impossible.*