HUTER
The Human Uterus Cell Atlas

Project Acronym:	**HUTER**
Project Full Name:	**Human Uterus Cell Atlas**
Call identifier:	H2020-SC1-2019-Single-Stage-RTD
Topic:	SC1-BHC-31-2019
Grant Agreement No:	874867
Start date of Project:	01/01/2020
Project Duration	2 years (24 months)
Document due date:	31/12/2020
Submission Date	19/01/2021
Leader of this report:	BAHIA
Deliverable no:	7.1
Deliverable name:	Final design of HUTER platform architecture
Dissemination level:	**Public**

Version History

Version	Date	Details
1.0	11/01/2021	Final version of platform architecture design. Deliverable completed.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 874867.

DELIVERABLE 7.1

Final design of HUTER platform architecture

Table of Contents

List of Acronyms

AI	Artificial Intelligence		**API**	Application programming interface		**AWS**	Amazon Web Services
BAHIA	Bahía Software S.L.U.		**CPU**	Central processing unit		**CZI**	Carl Zeiss Image format
DEV	Development		**DICOM**	Digital Imaging and Communication On Medicine		**EC**	European Commission
EC2	Elastic Compute Cloud		**ECR**	Elastic Container Registry		**eCRF**	electronic Case Report Form
ECS	Elastic Container Service		**EFS**	Elastic File System		**EKS**	Elastic Kubernetes Service
ELK	Elastic Stack		**GATK**	Genome Analysis Tool Kit		**GDPR**	General Data Protection Regulation
HCA	Human Cell Atlas		**HCA-DCP**	Human Cell Atlas - Data Coordination Platform		**HTTP**	Hypertext Transfer Protocol
HUTER	Human Uterus Cell Atlas		**IAM**	Identity and Access Management		**IT**	Information Technologies
JSON	JavaScript Object Notation		**JSP**	JavaServer Pages		**JWT**	JSON Web Token
NEMA	National Electrical Manufacturers Association		**PACS**	Picture Archiving and Communication System		**PRD**	Production
RDS	Relational Database Service		**REST**	Representational state transfer		**S3**	Simple Storage Service
scRNAseq	Single cell RNA sequencing		**SDK**	Software Development Kit		**SOAP**	Simple Object Access Protocol
SSO	Single Sign-On		**TSV**	Tab-separated Values		**VCF**	Variant Call Format
VPC	Virtual private cloud		**WDL**	Workflow description language			

1. PURPOSE OF THIS DOCUMENT

The purpose of this report is to present the final design of the HUTER Platform architecture, an evolution of the previous version reported in deliverable D2.1 of month 9 (document entitled "*HUTER_WP2_D2.1_Platform_architecture_design*").

The HUTER platform aims to support the research activities related to uterus cell map as well as other ambitious objectives such as develop a platform applicable not only in the context of uterus research (e.g. severe preeclampsia) but also adaptable and usable for other cases or diseases. As research activities have been developed, new requirements and specifications for the HUTER platform have been arisen which are collected in this report. Thus, the present document contains upgrades and improvements with the aim of being fully aligned with the last advancements of the project. The main changes have been mostly based on partner's progress in data collection, metadata and pipelines definitions as well as the feedback provided by HCA in our meetings with the aim of guaranteeing data and metadata transferring between both platforms (HUTER and HCA-DCP). Furthermore, some technical features briefly defined in deliverable D2.1 document are even further detailed and explained in this report. However, we want to highlight that the major content of the previous version, deliverable D2.1, remains in this report in order to offer the reader a better understanding of the whole HUTER architecture design.

The final platform architecture design reported herein is one of the key deliverables of WP7 "Platform integration" defined in the Grant Agreement. This document has been developed by BAHIA as lead of WP7 Platform integration, in close collaboration with the rest of the HUTER partners involved in this work package (INCLIVA, GRL, UPPSALA, CCHT and UEA).

1.1. Related documents

Documents linked to current actions to be delivered:

HUTER_WP2_D2.1_Platform_architecture_design

HUTER_WP2_D2.2_Deployment_of_HUTER_cloud_infrastructure

HUTER_WP2_D2.3_Beta_version_of_data_access_tools

Documents linked to future actions to be delivered:

HUTER_WP2_D2.4_Final_implementation_of_data_access_tools_and_DICOM_visualization_tool

HUTER_WP7_D7.2_Visual_System_implemented_&_Digitalisation_software_to_transform_images_from_research_equipment_under_opensource_standards

Other documents referenced:

HUTER_WP1_D1.1_Ethics_Plan

HUTER_WP9_D9.2_Data_Management_Plan

2. EXECUTIVE SUMMARY

HUTER project is aimed to create the reference cellular map of the human uterus, being involved in the international Human Cell Atlas initiative. In the HUTER project context, it is planned the development of an advanced platform to provide not only hosting and processing data features for cell sequencing and advanced microscopes techniques but also other functionalities to satisfy the specific requirements of HUTER researchers in terms of collaborative work and advanced tools. At the end of the project, the final platform generated during the HUTER project is expected to be applicable not only in the context of uterus research but also adaptable and usable for other cases or diseases.

One of the key parts of the HUTER platform is its cloud-infrastructure. This document delivers the final version of the HUTER platform architecture design. In this line, the previous delivered version of the architecture design (deliverable D2.1) has been even further defined and adjusted with the aim of being totally aligned with the last advancements of the project and its objectives. These advancements are related to the last progress in researchers' tasks, such as raw data collection, metadata and pipelines definitions as well as to feedback obtained from HCA-DCP coordinators through several technical meetings so as to guarantee a correct data transfer between both platforms.

As was reported in deliverable D2.1 report, the final architecture design has also considered not only HCA-DCP design guidelines but also HUTER specific requirements. Some of these general requirements are to facilitate collaborative work between partners, use state-of-the-art open technologies and foster the use of standards to represent and manage information generated in the project. All requirements must be implemented under a proper technological framework that assures the protection, safety and confidentiality of information. Considering all these aspects, 6 general components were designed to construct the architecture system with their interactions: Data Ingest, Storage, Processing, Data Access, Security, management and support layer and Dissemination & Communication components.

According to this functional division, a subsystem design of each general component was further defined including justifications of each decision taken with the aim of satisfying the HUTER requirements in the best manner. We argue not only the use of selected standards and technologies but also the deployment infrastructure regarding data volumetry, data nature (image, OMIC data and other data formats are expected) and collaborators laboratory workflows. Furthermore, some first approaches related to the integration with external applications, such as an early definition of APIs functionalities briefly introduced in deliverable D2.1 report were fully defined.

3. INTRODUCTION

3.1. HUTER project

The Human Cell Atlas initiative (HCA) aims to create molecular reference maps of all human cells to pool and expand knowledge of the diverse cells found within the human body in order to better understand human health, but also to improve diagnosis, monitoring and treatment of diseases. As part of the HCA, the HUTER project is focused on creating the molecular reference map of the human uterus.

This ambitious reference map will be developed thanks to applying state-of-art cell sequencing technologies to thousands of human uterus cells that generate vast amounts of diverse molecular data (order of thousands of gigabytes). Such amount of data makes difficult their processing, managing, hosting and sharing using traditional equipment such as common computers or servers. Consequently, the need of a powerful digital platform with advanced technical characteristics adapted to these challenges becomes in an unavoidable requirement to achieve the goals of the project.

In this context, several world research leader institutions are working together to set up an open cloud-based Data Coordination Platform (HCA-DCP) to check, share and analyse the data to be generated under the HCA, with funding and engineering support from private foundations. This platform will gather all the data generated in all HCA projects, such as HUTER, and share it openly with the rest of research community aiming to help generate the reference cellular map of human being. However, HCA-DCP is currently under development and the timing is beyond our control. Additionally, personal data collected from European participants involved in HUTER project must comply with the new European GDPR legislation. The EC is working with the HCA leadership to understand how to ensure HCA-DCP comply with new privacy laws in Europe. Until then, HUTER researchers cannot share their data collected with HCA-DCP.

In order to guarantee our competence to meet our project deliverables in due time and overcome the provisional regulatory obstacles, BAHIA in close collaboration with HUTER partners is implementing and developing a cloud-based hosting and an advanced platform with state-of-art functionalities and modules for the HUTER project, as part of contingency measures included in the Grant Agreement.

The HUTER platform and cloud will be active during all HUTER project. Therefore, the strategy to preserve all data hosted in HUTER cloud beyond the duration of the project complying with HCA sharing rules is to transfer the data gradually from HUTER cloud to HCA-DCP when its construction is completed and the data sharing is authorized. It must be highlighted that the transference of data between platforms is not a trivial question due to data hosted in HCA-DCP will come from not only HUTER project but also many other HCA projects that will have different kind of data, metadata, pipelines, etc which hinders the joint integration of all data. HCA-DCP managers have created open forums to define standardized metadata and technical aspects to overcome this challenge. In order to guarantee the compatibility and the data migration process between the HUTER platform and HCA-DCP, we are actively following and participating in the HCA-DCP forums and meetings to follow their guidelines and also to report HCA-DCP developers any technical peculiarity required for HUTER (e.g. custom metadata). Indeed, we have increased our communications through direct video-meetings with Data Wranglers of HCA-DCP (including coordinators of HCA-DCP) in order to discuss in depth important issues such as metadata compatibility, data formats, data transfer between both platforms, etc. All this feedback has allowed refining the HUTER platform architecture design with the aim of being fully aligned with HCA-DCP. Furthermore, this alignment between platforms and requirements has involved not only intensive coordination works but also an intern harmonization effort. The feedback obtained during the alignment meetings has provided us information to select and define metadata, pipelines, formats, and other important features properly.

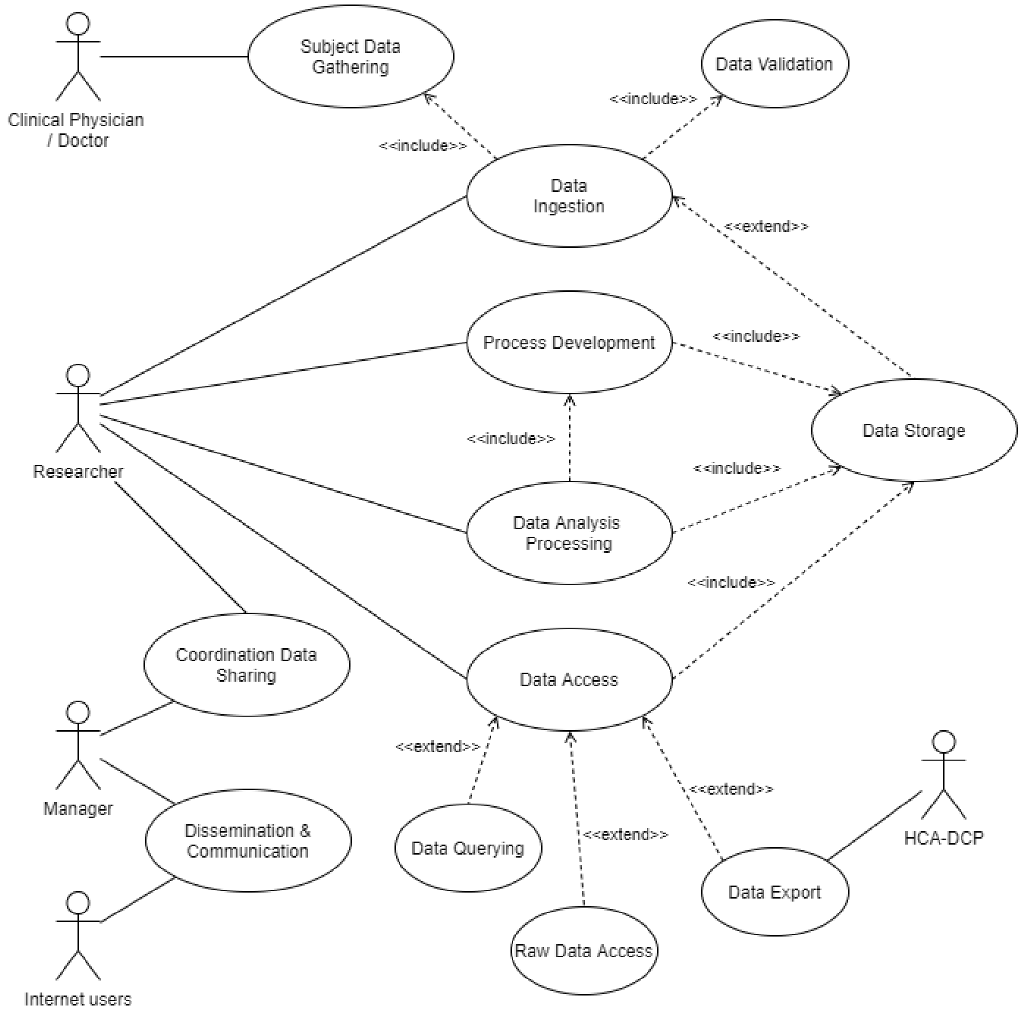
This deliverable D7.1 report defines the final version of the HUTER platform architecture design. The first version included in deliverable D2.1 document was taken as the basis to include new refine adjustments and to even further define some of the technical components based on the last progress in researchers' tasks such as data collection, metadata or pipelines definitions as well as our discussions with HCA-DCP staff during these three last months. Consequently, much of the content previously reported in deliverable D2.1 remains in this document with the aim of offering the reader a better understanding of the whole final design, without passing over any detail. In fact, it has been included a guide of changes in section 4 in order to identify the main changes easily.

3.2. Requirements

Since objectives of the HUTER project remain unaltered, main functionalities provided by the HUTER platform are the same ones that were presented in the previous deliverable D2.1 report. Just few features have been modified owing to data processing optimization or decision making about pending solutions.

In order to ease the understanding of this document section 3.2.1 *Functional requirements* has been added to remember provided functionalities by the HUTER platform as well as expected actors who will use them. Given that this section was deeply detailed in deliverable D2.1 document, a summarized description of these requirements will be shown.

3.2.1. Functional requirements



Picture 1- HUTER Platform context diagram

- **Subject Data Gathering:** This feature allows HUTER data collectors to register and manage electronically all the clinical information regarding subjects under a defined protocol.

  As data collection is performed by HUTER partners from different European countries a software solution has been deployed to ensure data homogeneity and ease the data entry process.
- **Data Ingestion:** This functionality provides the unique data entry point to the platform facilitating the control of the authorized data submissions. It also means the integration layer with HUTER partner's lab infrastructure that allows for an early data indexation in the platform in order to be accessible through deployed data access tools.
- **Data Validation:** Data Validation provides information check functions assuring the quality and integrity of submitted data.

- **Data Storage:** A shared storage system backs up HUTER researchers to release the collected data with all the partners during the project life. So storage is the addressed component for the information safe-keeping and communication.
- **Process Development:** Researchers and bioinformaticians are given a suite of tools that allows them to create, share and modify analysis process definitions by using a widely spread scripting language.
- **Data Analysis Processing:** Related to the Process Development, researchers and bioinformaticians are also supplied with the required infrastructure to run analysis processes over sample data using or not their own developed processes.
- **Data Access:** Searching and visualization tools are needed in order to allow for accessing indexed data in a proper way, that means, assuring authorized access and resources consumption.
- **Data Querying:** This feature is part of the Data Access functionality that allows users to browse the indexed metadata so as to localize raw data files.
- **Raw Data Access:** This is also part of the Data Access functionality that complements Data Querying with the capability of visualizing the information from the storage system. It is expected to need a specific tool for each type of data format so as to interpret its content.
- **Data Export:** According to the HUTER proposal, a HCA-DCP alignment is required in order to eventually move results to its platform. HCA-DCP guidelines are still under construction so the HUTER platform will use an own data model that will be translated to the final HCA specification when its definition fits the HUTER metadata needs.
- **Coordination Data Sharing:** As for HUTER project management, a common communication channel for partner’s teams is needed so the HUTER Platform provides tools to keep collaborators in touch and coordinate efforts.
- **Dissemination & Communication:** Among HUTER project agreements there is the requirement of publicizing project activities and results. With the purpose of ease this task, the HUTER Platform offers mechanisms for advertising HUTER activities on the internet.

3.2.2. Platform transversal requirements

In addition to functional requirements some mandatory features that affect to every component in the platform were identified and detailed in the previous deliverable D2.1. To make easier the understanding of this document, a brief summary of the transverse requirements which are obeyed in the HUTER platforms is shown.

- **Anonymous traceability of the information:** The HUTER platform has to assure the capacity of track the relations among the stored data following rules defined in the HUTER Ethics Plan. According to this plan, de-identification of the participant subjects has been implemented by the assignment of a study code.

- **Data Security:** The HUTER platform has to provide mechanisms that prevent from intentional or accidental destruction, modification, exposure or misuse of data.
- **Access Control:** Despite data anonymity, the HUTER Platform has to provide mechanism for access controlling and data obfuscation in order to protect data against unauthorized access**.**
- **Location Independent Access:** Since HUTER project is a collaborative effort between 6 European institutions located in different countries, the HUTER Platform has to be accessed from any collaborator's location.

4. CHANGES GUIDE FOR THE READER

As previously commented, this document contains the final version of the HUTER platform design. However, the major content of the previous version (deliverable D2.1) remains in this report in order to offer the reader a better understanding of the whole HUTER architecture design. We have included changes and upgrades based on partners progress in data collection, metadata and pipelines definitions as well as on the feedback provided by HCA in our meetings. Furthermore, some technical features briefly defined in the deliverable D2.1 document are even further detailed and explained in this report. In this section, we provide a changelog to identify the new adjustments of the architecture design easily.

SECTION TITLE	CHANGELOG
5 PLATFORM ARCHITECTURE	In section PLATFORM ARCHITECTURE a functional overview has been detailed in order to clarify the aim of each component. Albeit fringe of modules remains unchanged, some inner components have suffered from modifications. In further sections, added and modified components will be technically explained.
5.1 Data Ingest	This module has suffered from modifications aimed to reduce the uploading component dependencies and complexity. So ingestion manager has been also modified to add the deleted functionality from the “uploader”.
5.2 Storage	In this module two new functionalities have been added. One of them is a technical decision for process optimization and, the other one is a new feature that allows the HUTER Platform to cover a new suggested requirement by the partners.
5.3 Processing	Just one new component has been proposed in this module as an independent service aimed to analyse DICOM imaging through Artificial Intelligence algorithm.
5.4 Data Access	After a requirement refinement made co-ordinately with the partners, a more detailed set of tools for data accessing has been proposed. In this section, there is an introduction to the suggested functionalities of each tool. The HUTER Platform is ready to integrate new tools depending on the feasibility of their

	integration capacities.
6.1 Security, Management and Support Layer	This section has been fulfilled by providing the tools that will support Microservice platform for the HUTER Project.
6.2 Data Ingest	Apart from detailing unanswered question in this section, some modifications regarding to Broker – Uploader have been explained as well as the full definition of the Ingestion Manager.
6.3 Storage	In order to provide a HUTER interface for storage access, Storage Manager has been deeply explained. Besides, a new functional storage has been introduced: Genomics Data Indexer.
6.4 Processing	An additional component for imaging analysis has been proposed: Imaging AI Server.
0 In Picture 16, components of Pipeline Web Manager are shown and next explained: • **Core:** the main logic is contained in a core that orchestrates the consumption of the storage manager services to retrieve all the required data. • **Pipeline Web Manager Portal:** it exposes a web interface integrated to HUTER SSO that applies a security layer. It also eases the access to data and metadata that can be analyzed. • **Cromwell Client:** it is a REST client of the Cromwell Server to request the execution of a pipeline over preselected data. • **Git client:** it means a client of the repository that allows the	This section has widely improved because a list of proposed Data Access Tools has been included as well as some DICOM Viewer modifications.

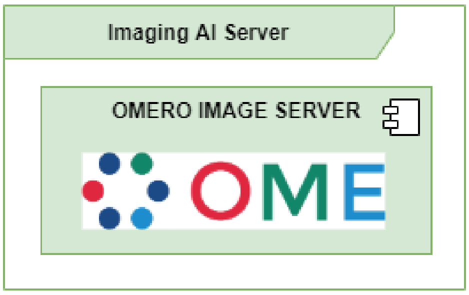
Pipeline Web Manager to retrieve the WDL definition from the git repository and send it to Cromwell through its client. • **Storage Manager Files Client:** it is a REST client to query the files indexed in the platform through the Storage Manager. This service will provide the location of the files to be processed in the pipelines as well as the registration mechanism for the new ones. • **Storage Manager Subject Client:** it is a REST client for subject metadata access. • **Storage Manager Samples Client:** it is a REST client that allows the access to indexed sample metadata and allows users to select the sample metadata files to be used in the analysis process. • **Storage Manager Processes Client:** it is a REST client that allows the pipeline web manager to provide the data tracking by registering each pipeline execution as a process in the platform as well as inputs and outputs.	

<table>
<tr>
<td>

4.1.1. Imaging AI Server

The Image AI Server was covered by the deployment of an OMERO Server. OMERO is a suite of tools for imaging analysis widely used in researching environments. It applies Artificial Intelligence algorithms on imaging data in order to detect some features in the images such as segmentation and region of interest detection.

The challenge of this integration is that the algorithms should be applied on DICOM images what involves managing the DICOM imaging peculiarities through an integration with de DICOM Viewer.



Picture 17 - Imaging AI Server

Data Access

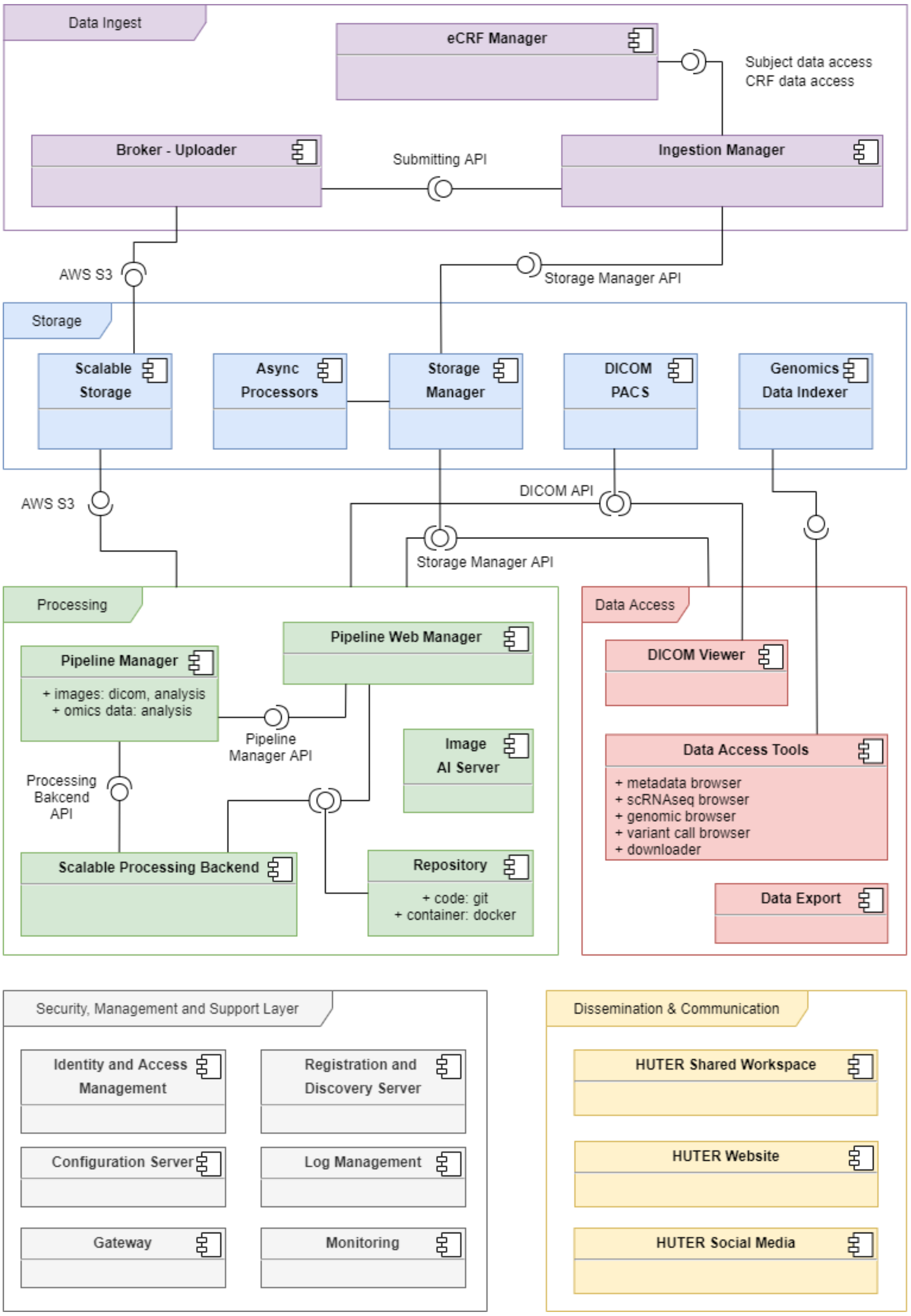
</td>
<td></td>
</tr>
<tr>
<td>7.1 Deployment model</td>
<td>The final version of the deployment infrastructure is shown in this section.</td>
</tr>
<tr>
<td>7.1.2 Application servers</td>
<td>Pending platform definition has been clarified with regard to the needed infrastructure for application servers and services deployment.</td>
</tr>
<tr>
<td>7.1.3 Databases</td>
<td>In order to complete the storage infrastructure, two new databases have been suggested: NoSQL and an ELK Cluster.</td>
</tr>
</table>

8.2 HCA alignment	In this section the IT requirements for a full alignment with the HCA are gathered together.

Table 2- Changelog of final version of the HUTER architecture design

5. PLATFORM ARCHITECTURE

An overview of main components in the HUTER Platform architecture will be shown in this section and modifications from deliverable D2.1 will be pointed out. The description of subcomponents will be included in section 6 *SUBSYSTEMS DESIGN DESCRIPTION* with the aim of giving a more detailed knowledge of subcomponents, especially about those new ones, clarifying both technical and functional reasons that support the decision.



Picture 3 - High level architecture diagram

A high-level diagram of the HUTER Architecture (Picture 3) shows designed modules as well as the main technological components that will be provided. The aim of the schema is to be used as an overview of the whole platform.

The HUTER Platform architecture keeps on covering functionalities for requirements listed in section *3.2 Requirements*. Current modifications just affect to additional features or to the decision of implementing alternative workflows to get required functionalities.

Therefore, main components will be briefly listed below, as point of reference from deliverable *D2.1*, but new ones or their modifications will be deeply detailed.

5.1. Data Ingest

In this module, data gathering functionalities are assembled to point out the interaction with subject and sample data collection. No additional components have been required so this module is still composed by three elements:

- **eCRF Manager**. This software is addressed to collect, check and export personal data about study subjects into the HUTER Platform. Its functionalities were no modified further than described ones in deliverable *D2.1*.
- **Broker – Uploader**. It is a component aimed to move raw data files from partner's laboratory storage to HUTER Platform assuring the data integrity and traceability. This final version of the architecture has been modified in order to delete the integration with the eCRF Manager. This communication was aimed to retrieve subject data gathered by clinical users but it has been moved to the Ingestion Manager which was already integrated with the eCRF Manager. In such a way, this new design provides less integration points and, hence, less dependencies of proposed components.
- **Ingestion Manager**. This component is focused on submission request management, what means requests of raw data ingestion, and the retrieving of the metadata from the gathering tools (eCRF Manager). This component is exposed as a set of services that work as an integration API. It also performs an early data validation that will be consolidated in the platform. Further ahead it will be technically detailed but it must be pointed out that, unlike its previous design showed in deliverable *D2.1*, Ingestion Manager has been changed to unite all the required metadata integration processes for the data submission. So this component performs the role of unique data entry point into the HUTER Platform.

5.2. Storage

The HUTER Storage module consists of a set of services that allows users to save sample data in a shared system while it assures the tracking of relations among stored data and metadata.

This module has changed from its version in deliverable *D2.1* with the addition of two components that provide new features meanwhile the rest of components previously designed remain the same. Albeit, they are shown below to provide a complete overview of the storage module:

- **Scalable Storage**. This component was presented in deliverable *D2.1* and keeps on offering a flexible disk space that can unlimitedly grow according to the researchers needs. It is aimed to save raw data files no matter their format.
- **Storage Manager**. This component is composed of services that work as an API to make possible the storage, enquiry and modification of the indexed data and metadata. The inner composition of this API will be later explained in section *6.3.2 Storage Manager*.
- **DICOM PACS**. Additionally to the raw data storage, the HUTER Platform provides a DICOM Storage system, also known as PACS, for managing images in DICOM format. Not only PACS performs storing functions but also offers a DICOM compliant interface to query, upload and retrieve DICOM imaging.
- **Asynchronous Processing**. This new component is a solution to attend high-cost processing requests from the ingestion API avoiding long response delays to the clients. It was arisen by the need of decoupling the submission reception and the data validation applied to the raw data. Albeit, it could be used to execute other heavy tasks related to the data indexation.
- **Genomics Data Indexer.** Regarding new requirements suggested by the HUTER researchers, this new proposed component is a complementary and temporary storage for data generated in a whole genomic sequencing process. Genomics data is presented as extremely large files that usually contain character sequences like ones in VCF file format. Over this information, researchers have pointed out the possible need of executing queries and comparisons among different files in order to detect genomic variants. Further ahead, in section *6.3 Storage*, the technical implementation of this component will be detailed to argue how to face the challenge of nimbly managing this information format.

5.3. Processing

Processing module is the set of tools deployed in the HUTER Platform so as to define and execute heavy processes over stored data. It also provides mechanisms to consolidate the resulted data in the platform in order to trace the relation between process inputs and outputs. The flexibility of this module enables the execution of a wide range of processes such as omics data analysis, data format transformation, imaging analysis and so on.

In deliverable *D2.1*, this module was introduced in order to support omics data analysis as well as to implement the transformation of electronic microscope imaging into DICOM format. However, in this document a new component for DICOM imaging analysis through Artificial Intelligence algorithms has been considered. Although the initial design of the module allowed for AI imaging processing would imply the development of AI algorithms. To avoid this additional effort, a more convenient solution was chosen by deploying an existing tool that provides all the imaging analysis functionality.

Likewise, this module offers the possibility of creating new process, called pipelines, and defining execution environments for them. Researchers can get advantage of it to develop new ways of treat data by defining pipelines in a standard language supported by HCA.

The final set of components of this module is:

- **Repository**. It consists of tools for storing and sharing resources among collaborators. Specifically, the HUTER platform defines two types of repositories: code and execution environments. Hence, collaborators could work together in the definition of new analysis methods or data transformation ignoring the location of the rest of their workmates.
- **Pipeline Manager**. This component is an orchestrator for the execution of processes in a scalable backend. So it is in charge of provisioning the infrastructure to execute the requested tasks and managing the input and output data involved in the process.
- **Pipeline Web Manager**. This component offers a user-friendly interface that allows researchers to collect all needed resources from the HUTER Platform so as to invoke the Pipeline Manager. Additionally and equally important "Pipeline Web Manager" provides a secure layer that prevents users from mixing unrelated data during processing solicitation.
- **Scalable Processing Backend**. For the purpose of executing pipelines a flexible environment is needed. Scalable Processing Backend consists of a dynamic cluster which allows for running pipelines in predefined execution environments. These ones can be created on demand depending on the number of received execution requests. Each of these environments is aimed to provide the infrastructure and tools required by a specific pipeline.
- **Imaging AI Server**. DICOM imaging analysis has been proposed for complementing the DICOM Viewer features with AI imaging analysis. The main reason for defining an independent AI component of common pipeline execution was the possibility of integrating well known systems that already exists in the researcher's community. Even though there is no limitation defining imaging analysis as pipelines, the current proposal offers a more mature and reliable solution. This approach

also explores a new bet for applying AI algorithms over DICOM images what is not deeply spread in DICOM community.

5.4. Data Access

Data Access module covers all functionalities that allows users to access the stored data in the HUTER Platform. It consists of distributable or web-based tools for data querying, retrieving and visualization.

Specific tools are being developed so as to manage the different data types, like DICOM Viewer for DICOM imaging as well as Data Browser for metadata discovery. Moreover, as for following HCA DCP guidelines, a helping tool for data exporting to HCA Platform was added to this module in order to get a full alignment with the HCA DCP requirements.

- **Data Access Tools**. In an early approach, this component was expected to be a set of tools aimed to visualize stored data and metadata from the platform. After these months of cooperative work with partners about requirement refinement, a more detailed set of tools has been proposed in order to deal with each type of data format. In this line, this architecture will support access tools not only for data generated by state-of-the-art cell sequencing techniques (e.g. scRNAseq) but also for classical genomic data (obtained by traditional whole genome sequencing techniques). This will provide a further complete platform at the end of the project. Next, a list of proposed solutions is introduced to cover arisen areas of interest for researchers:
    - **Metadata Browser:** regarding to the essential need of finding information in the platform, this component allows for querying the stored metadata. With the help of this tool, users can search metadata applying filters related to subject, sample, file or process characteristics. Likewise, users will need this tool to find the location of the stored files in the platform so as to be able to download or launch additional tools over them.
    - **Downloader:** this component works as a complement to Metadata Browser allowing HUTER authorized users to download raw data from the Storage Module.
    - **scRNAseq Browser:** HUTER partners have revealed that a tool for Single Cell RNA Sequencing data interpretation would be useful for this project. According to them, we agreed to evaluate the integration of a tool to cover this need. Since the implementation of this tool means a project itself, the deployment of an already existing tool has been considered in order to focus the effort on adapting it to the HUTER Platform.
    - **Genomic Browser:** According to HUTER researchers working on genomic data, a tool for genomic data visualization could also be useful. It has been evaluated the possibility of integrating an existing tool aimed to visualize genomic sequences.

- **Variant Call Browser:** In previous sections, it was mentioned the likely need of browsing genomics data for specific gene sequence variations among data of different samples. In this context, an early version of such a tool will be explored. Due to this proposal, in section *5.2 Storage*, a new component for omics data indexation has been included.

- **Data Export**. Thanks to a coordinated work between HCA-DCP (Data Wranglers) and HUTER partners, the requirements of this early expected component have been drafted. As for moving project data to the HCA platform, it would be convenient to have a tool for the translation of the HUTER metadata format to the HCA one. Due to HCA model must be adapted to some requirements of HUTER project such as new data fields or the enrichment of current HCA ontology; this tool will be developed in a late step.

  Furthermore, it could be considered the inclusion of integration functionalities through HCA API depending on the HCA system maturity and HUTER agreements.

- **DICOM Viewer**. As for the proposed DICOM imaging solution, in deliverable *D2.1* was introduced the deployment of a DICOM Viewer. This component will allow users to visualize images from private scanner imaging formats previously transformed to DICOM with the help of a pipeline. This solution also requires the mentioned PACS Server and will be boosted by the integration with a dedicated server for Artificial Intelligence imaging analysis such as segmentation or region of interest detection.

5.5. Security, Management and Support Layer

This module involves several features that any IT system needs to cover transverse requirements related to Data Security, Access Control and Communication Management.

No new functionalities have arisen so next listed components are retrieved and summarized from deliverable D2.1:

- **Identity and Access Management**. This component manages user accounts for all the components in the platform providing the Single Sign On (SSO) functionality. Thanks to that, a common security layer has been deployed over the whole platform.
- **Log Management**. In order to centralize the access to the log files of every application that compose the HUTER Platform, a tool will be deployed to gather those logs and show them in a common console. This also provides a powerful tool to keep track of any error that the platform could suffer.
- **Monitoring**. Although this component is closer to deployment infrastructure is important to point it out here. It will provide a console to watch Platform health status so as to obey availability requirements.

- **Microservice Platform**: In order to deploy the service layer (API) so as to support the data consumption and enrichment in the HUTER platform, a Microservice architectural solution was chosen. Thanks to that, the HUTER Platform will be able to attend a high-load of service requests by the automatic creation of service instances depending on the demand. There is a large list of tools to implement this approach which is widely deployed all around the world. Likewise, most of these tools rely on the next components that were previously mentioned on deliverable *D2.1*:
    - **Registration and Discovery Server**. This component is required in order to provide scalability to the designed service layer following a microservice approach. Due to its knowledge of deployed services in the platform, it helps to bear load of requests by locating any service instance that is managed in the Microservice Platform so as to redirect the request to it.
    - **Gateway**. This component means the unique entrypoint to communicate with the HUTER API form data accessing. On any distributed system, a gateway is needed to provide a reachable access to underlying service instances. It also facilitates the infrastructure configuration and load balancing by hiding the infrastructure complexity to the clients with the help of the Registration and Discovery Server.
    - **Configuration Server**. In order to ease the service layer configuration, this component allows administrators to change configurations of every instance of a service from a centralized point. That is really useful based on the fact that service instances are created on demand depending on the platform load so their manual or static configuration would be unacceptable.

No functions have been added to this module because the previous proposed ones have covered all the requirements. However, it must be remarked that in section *6.1 Security, Management and Support Layer* this module will be detailed to identify the final solution which was pending to be defined in deliverable *D2.1*.

5.6. Dissemination & Communication

This administrative and communication module is composed of tools that facilitate team managers the spreading of HUTER information among their teams and internet users. As this module has suffered from no modification, just a brief description of each component has been retrieved form deliverable *D2.1*:

- **HUTER Shared Workspace**. Platform provides an internal networking tool to allow team members to share management information as well as to be used as a virtual desk.
- **HUTER Website**. A website has been created in order to publish HUTER project progress on the internet. This web offers a public area in which projects targets, participant introduction and news

are shown to internet users. It also offers a private area that helps SSO authorized users to access to the HUTER Platform tools.

- **HUTER Social Media**. Although these components are not deployed on HUTER infrastructure, they have been registered here to point out the effort of team managers in the HUTER project communication on several Social Media Platforms.

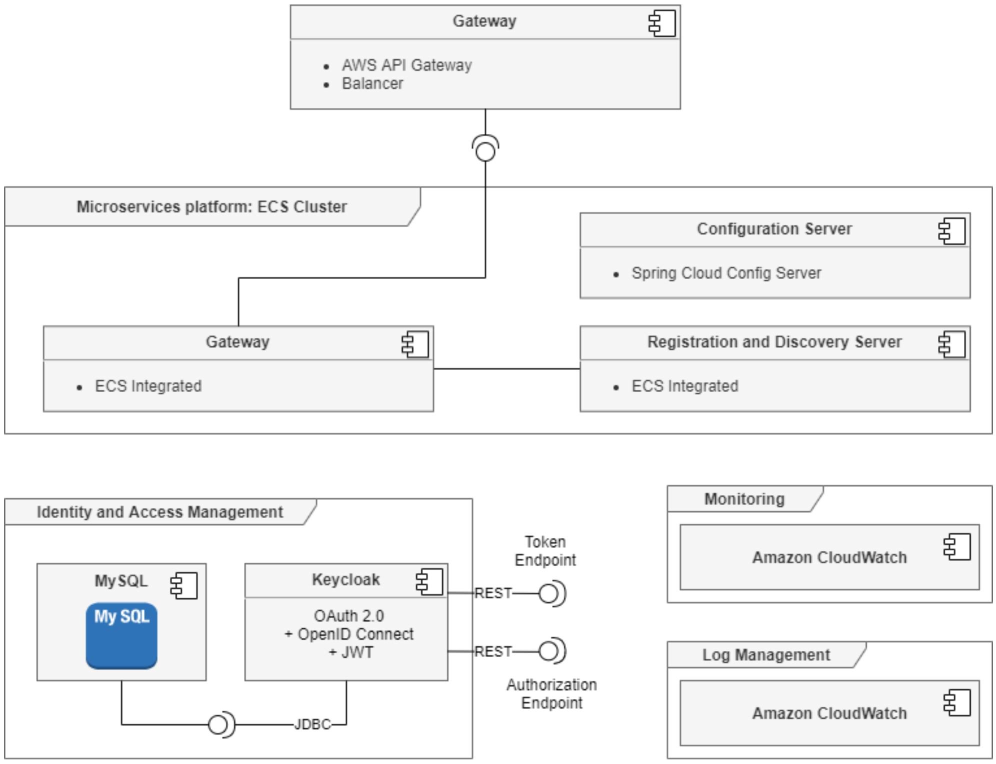
6. SUBSYSTEMS DESIGN DESCRIPTION

In this section a more detailed and technical explanation of the HUTER Platform Architecture will be shown focusing on new components of the HUTER Platform. However, a brief description of unaltered components will be given so as to clarify the final infrastructure.

The order of the modules presentation has been altered from previous sections because it will be clearer to understand the technical relations among modules following this order.

6.1. Security, Management and Support Layer

This module contains applications and tools that cover some of the project transverse requirements which affect to all components. Its components are mainly focused on providing homogeneous methods for user authentication, platform monitoring and service deployment.



Picture 4 - Security, Management and Support Layer

Unlikely deliverable *D2.1*, Picture 3 shows a more detailed implementation of the components listed in section *5.5 Security, Management and Support Layer*. Although some of them were already defined and remains the same, other components needed a complete definition. In the next section, those uncompleted ones will be pointed out.

6.1.1. Identity and Access Management

In deliverable *D2.1* the implementation of this component was defined by the integration of a Keycloak Server. This solution is still valid and allows HUTER Platform to provide a SSO layer for access control applying a combination of OAuth2.0 authorization and OpenID Connect authentication. Besides, the communication is made through JWT (JSON Web Token) what simplify the integration of this security layer in clients through a widely and reliably open source implementations.

6.1.2. Log Management

As for log management, Amazon Cloudwatch service was chosen due to its ease integration with the deployment infrastructure what requires no additional effort beyond a configuration process.

6.1.3. Monitoring

Amazon Cloudwatch also provides monitoring features so it was chosen as the tools for this purpose.

6.1.4. Microservice platform

In order to provide a scalable service layer that supports the proposed API, a microservice solution was drafted in deliverable *D2.1*. This approach allows for auto-deploying service instances depending on their demand to provide a high available system. The structure of this platform in order to correctly implement the microservice approach will need an inner Gateway, a Registration & Discovery Server and a recommendable Configuration Server.

In the previous deliverable *D2.1* several possible tools were listed to implement this platform however the final option was not among them. Getting advantage of the AWS infrastructure, a service called ECS (Elastic Container Service) has been deployed to provide an almost complete solution for the microservice deployment. ECS allows for defining service clusters assuring their availability with a little management effort. ECS provides out-of-the-box features that perform as Gateway and Registration & Discovery Server avoiding the need of implementing and administrating these tools.

If a Configuration Server was required, it should be deployed like any other service in the ECS what has been done with a Spring Cloud Configuration Server using the existing HUTER git service as the configuration storage.

The unique requirement to be taken into account for ECS exploitation is that services have to be released inside docker containers.

6.1.4.1. Gateway

The gateway was aforementioned in deliverable *D2.1* as the unique entrypoint to reach the services and tools deployed in the HUTER network. However, to clarify the Gateway schema in Picture 4 - Security, Management and Support Layer, both drew “Gateway” components will be explained.

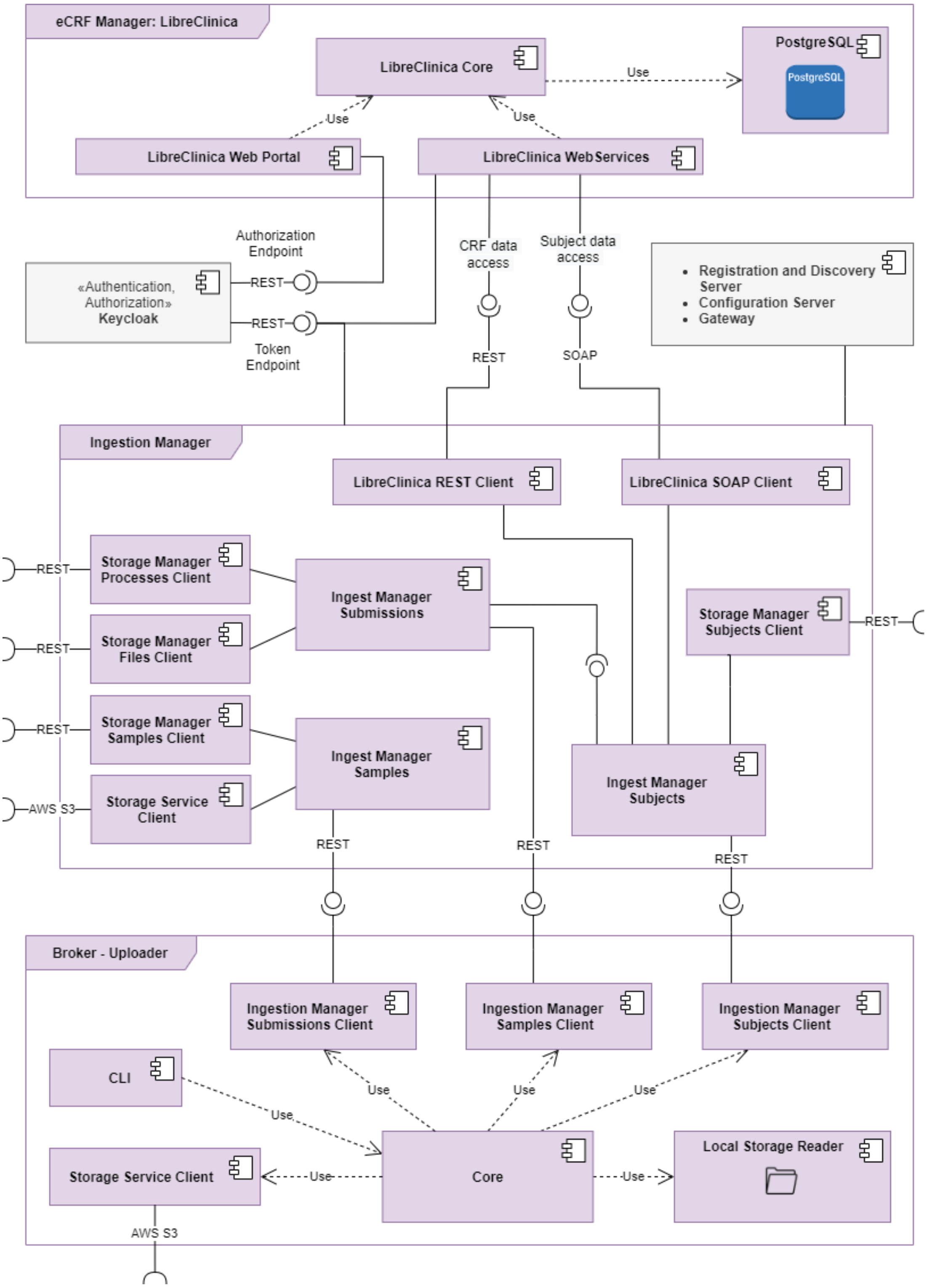
A main gateway allows user requests to access to the internal network of the HUTER platform. This one depends on the deployment platform, as it is exposed in section *7.1 Deployment model*, so two AWS Services are in charge of this functionality. The very first subcomponent is the AWS API Gateway which publishes an entrypoint to the internet. Among its functionalities, it is responsible for request routing by domain policies given that the HUTER Platform is composed by two different environments: DEV for application testing purpose and PRD for exploitation by users. The second subcomponent is a HTTP Balancer that allows any request to reach the right “machine” inside the environment.

A secondary gateway was mentioned in deliverable *D2.1* in order to manage requests towards the microservice platform but it has not defined until now. This independent gateway is required in a microservice approach to deal with the auto-scalability of this kind of systems decoupling the service client from the service endpoint location. The microservice gateway is integrated with the Registration and

Discovery Server in order to ask for the current location of a free service instance that can attend each request.

6.2. Data Ingest

In Picture 5, a detailed design of the data ingestion module is shown. Component boundaries and published services have lightly changed from the drafted version in deliverable *D2.1*. This new version of the schema describes the current composition of the module so as to support the data gathering workflow of the HUTER partners. This module fits the data gathering protocol that was defined in the HUTER project proposal. Furthermore, it is also adapted to the laboratory workflow in order to facilitate the sample data uploading to the HUTER platform without interfering the common activity of the researchers.



Picture 5 - Data ingestion design

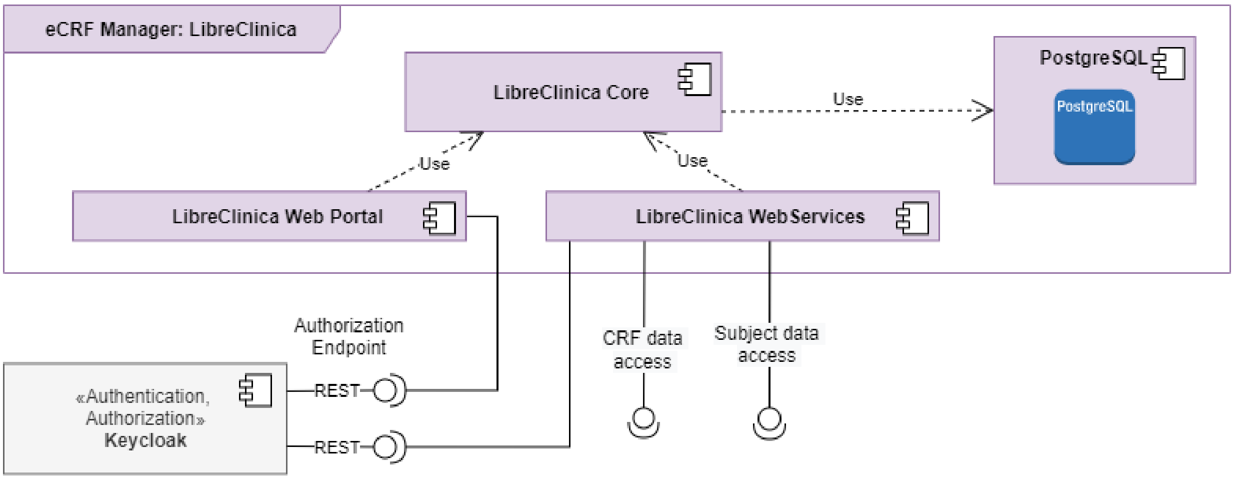
6.2.1. eCRF Manager

The LibreClinica implementation of the eCRF manager has not changed from the first proposed one in deliverable *D2.1* and it is sketched in Picture 6. However, the final deployed version has been completely updated with a new user-friendly interface and some ad-hoc features for HUTER users.

The inner structure of LibreClinica consists of:

- **PostgreSQL database:** for data persistence and to manage the information life-cycle in the application.
- **LibreClinica core:** this subcomponent contains the logic of the application that can be invoked from either the web portal or the web services.
- **LibreClinica Web Portal:** it is an old-fashion but reliable Java web application made with Spring 3, JSP and Javascript. The web interface has been updated by Bahia Software using Bootstrap 4 and a new version of JQuery in order to make it more usable. It has also been integrated with Keycloak in order to get advantage of the Single Sign On.
- **LibreClinica Web Services:** it is a service interface composed by SOAP and REST-like services for different kind of information enquiry. In the HUTER project, SOAP Services will be used to access to subject information in the HUTER study meanwhile REST services will be used to retrieve the data registered in CRFs.

  Both kinds of services have been integrated with Keycloak to apply a unique control on the access requests.



Picture 6 - LibreClinica subcomponents

6.2.2. Ingestion Manager

This component has been designed to perform a gatekeeper and integrator role. No data submission will be allowed unless Ingestion Manager has granted access after a protocol execution. It also performs integration tasks depending on the source of the information, in the HUTER project, the LibreClinica eCRF manager.

The implementation of the Ingest Manager is a layer of services which is published as a REST API in order to be consumed by the data uploaders. Albeit, it must be highlighted the existence of a specific interface for raw data uploading that depends on the storage module, in this case: AWS S3.

As it is seen in Picture 7, Ingestion Manager provides the promised REST interface which exposes microservices in order to manage data submissions to the HUTER platform. It is also integrated with other HUTER platform components through clients under different protocols such as REST for LibreClinica and Storage Manager API or SOAP, also for some LibreClinica integrations.

In this document, a more detailed composition of this solution is shown so the detailed explanation of each subcomponent is done next:

- **Ingestion Manager Subjects:** this service is published under a REST approach using the mentioned ECS microservice platform. The aim of this service is to safely gather the subject data from the eCRF Manager and store it in the HUTER Platform through the Storage Manager Service. It also offers methods for subject data querying that will be redirected to the storage manager API.

  To clarify the aim of this component the explanation of its integrations will be detailed:

  - LibreClinica SOAP Client: this client allows the ingestion manager subject service to retrieve subject data from LibreClinica in order to validate the existence of the subject in the study.
  - LibreClinica REST Client: LibreClinica publishes CRF data of each subject through a REST Service. That information is considered the metadata of the subject in the HUTER project so it must be retrieved and consolidated in the HUTER platform.
  - Storage Manager Samples Client: the unique way to access to the HUTER Platform database will be through the storage manager API what means a decoupled layer for data access. Therefore, this service needs a Storage Manager Subjects Client to save the subject data previously retrieved and transformed into the HUTER JSON format.

- **Ingestion Manager Samples:** this service is analogous to the Ingest Manager Subject but it manages sample data for its transformation and storage as HUTER metadata. The source of the sample information is not LibreClinica but this data is provided by the researcher from laboratories as TSV files previously ingested. This workflow was agreed with partners for avoiding the inclusion of new steps in the labs protocols. Regarding this need, the ingestion manager for sample data will access to files store in AWS S3 to translate it into HUTER model and save it.

  To clarify the aim of this component the explanation of its integrations will be detailed:

  - Storage Service Client (AWS S3): sample data will be firstly stored in AWS S3 as TSV files like any other raw data. This will be done through the uploader identifying these files as METADATA. After that, Ingest Manager Samples will be able to access to these files through a client developed with the AWS SDK for S3 Service in order to parse those files and retrieve the data.

- Storage Manager Samples Client: according to the proposed approach, the unique way to access to the HUTER Platform database will be through the storage manager API. Therefore, this service needs a Storage Manager Samples Client to save the sample data previously gathered and transformed into the HUTER JSON format.

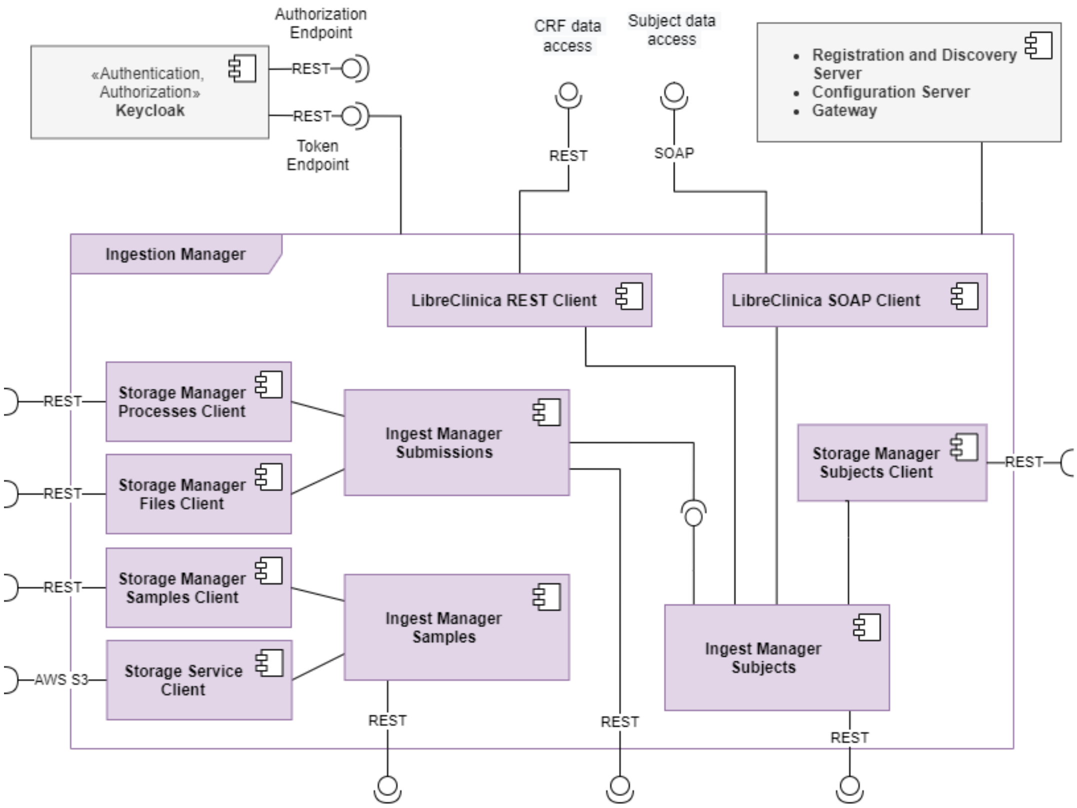
- **Ingestion Manager Submissions:** this service will provide the capacity of early register files in the platform. This does not mean that files will be uploaded through this service but a reference of their existence will be stored in the database as well as their current real status, for instance: validation pending, stored, validation failed, no file found or deleted.

  Submission requests will be done by the uploader after the upload of the required files. The information about those uploaded files will be sent to the Ingest Manager Submissions to create or update the registers in the HUTER database and their status. In order to do that, Ingestion Manager Submissions will need next subcomponents:

  - Storage Manager Processes Client: by using this client of the Storage Manager, this service will be able to keep tracking of all uploaded files. Ingestion Manager Service will use that client to create a submission process and link it to the file registers.
  - Storage Manager Files Client: this client will be used to store and update the information about each uploaded file in the platform decoupling the Ingestion Manager from the database implementation like the other services.

    This service will also be used to send validation requests for the ingested files. Even when the data validation process is part of the ingestion, the validation process interface has been deployed as part of the storage manager in order to maintain a similar interface to storage manager instead of one close to the client format. However, this decision could be reverted without impacting the functionality.

Finally, unlike deliverable D2.1, the direct integration of this component with the database containing all the metadata has been dismissed so as to provide a decoupled layer from the storage model.

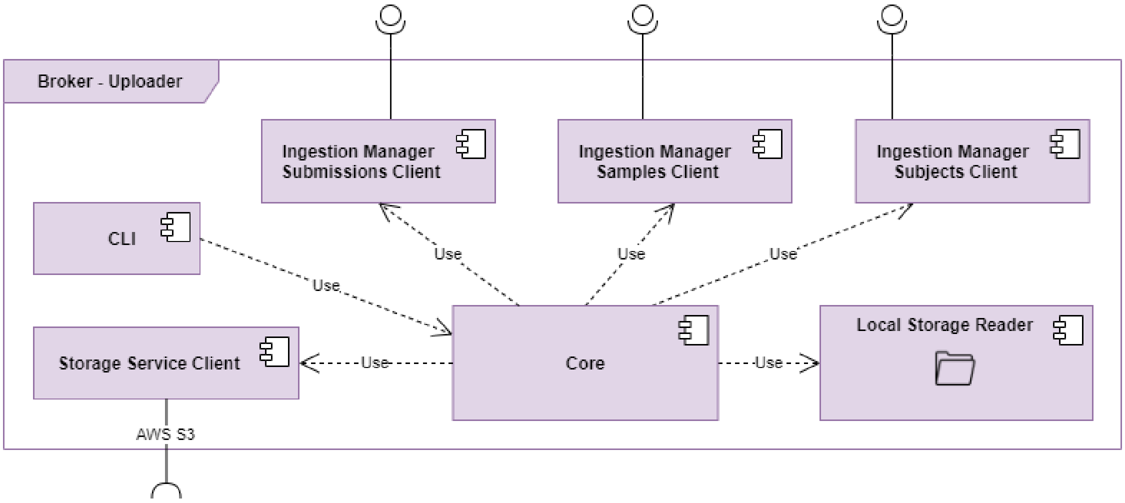


Picture 7 - Ingestion Manager subcomponents

6.2.3. Broker – Uploader

The Broker - Uploader tool showed in Picture 8 will be delivered as a *heavy* client to our partners so as to perform the upload of data and imaging files from labs infrastructure to the HUTER Platform. In this document the previously shown design in deliverable *D2.1* has been improved to deeply describe its composition.

As for the aim of this component, just one feature has been deleted: the integration with the eCRF Manager to retrieve CRF data which has been moved to the Ingestion Manager. Apart from that, this component keeps on being responsible for the raw data upload to the HUTER Platform supporting raw files of several types such as FASTQ, TSV, SVS, CZI and so on, depending on the contained information.



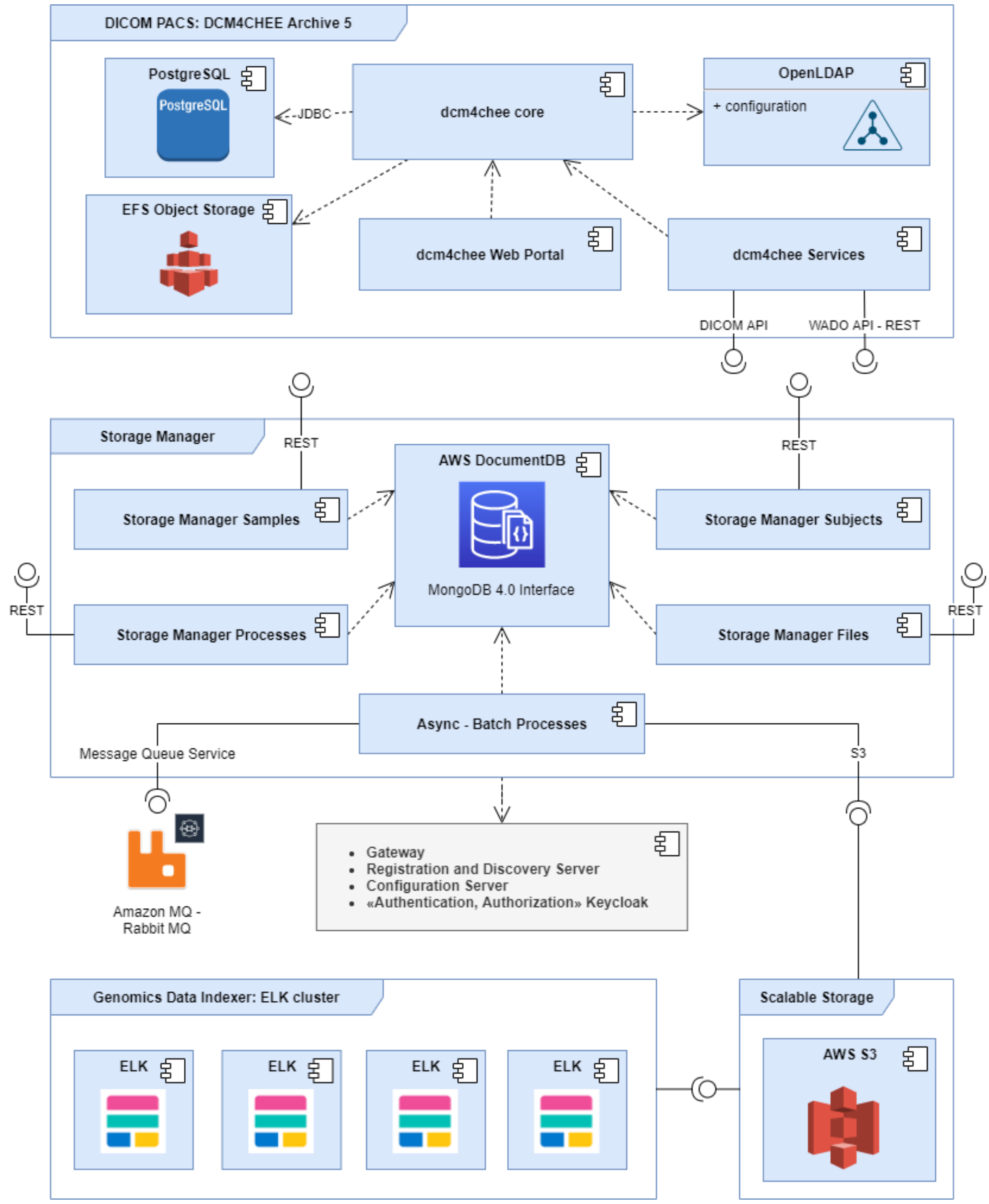
Picture 8 - Broker - Uploader subcomponents

To get a better understanding of how this tool works, a list of their subcomponent is provided next:

- **Core:** this subcomponent means the logical heart of the tool which orchestrates the workflow for files uploading and services invocation.
- **Storage Service Client:** this subcomponent is an abstraction that allows the uploader to access to a remote storage service and perform the file uploading. In our case, it has already been mentioned that the remote storage will be the AWS S3 Service. The implementation of this connector will be done by the AWS SDK Java library that provides mechanism to communicate with AWS in a secure way.
- **Local Storage Reader:** the source of the information will be partner's laboratory machines that will need the HUTER Uploader tool for data ingestion. So as to access the raw data files, Uploader tool needs a subcomponent to read files from a local file system.
- **CLI:** Command Line Interface will be the way of running the Uploader tool on PC or Mac. It is needed to be run on a machine with access to the local storage that contains the sample data files.
- **Ingestion Manager Subjects Client:** in order to ingest subject data, the Uploader will allow users to request this action to the Ingestion Manager Subjects server side.
- **Ingestion Manager Samples Client:** sample data ingestion will also be needed for later processing so the Uploader will allow users to request the ingestion of sample data from a previously ingested METADA file.
- **Ingestion Manager Submissions Client:** any uploaded file into the HUTER Platform has to be consolidated in the database through a submission process. Otherwise it will not be available for its treatment through HUTER tools like analysis pipelines or data access tools. This client will connect the Uploader tool with the server-side so as to send information about files that users would like to upload to the platform as well as their related metadata.

6.3. Storage

The storage module is responsible of data storing and indexation in a database for later enquiry. Both functionalities had been defined in the deliverable *D2.1* and they are also present in the final design as well as new functionalities that extend the first ones. All the final components can be located in Picture 9 - Storage module.



Picture 9 - Storage module

Apart from offering an unlimited cloud storage space and the indexation of these files in a database enriched by metadata gathered from other ingestion processes, the HUTER platform provides two new features in order to reinforce storage module:

- **Asynchronous processing:** this new feature has been added to avoid long time waits for service responses related to file validation tasks. It is based on an integration pattern that uses message queues to early response to clients as soon as their requests were stored for a later processing.
- **Genomics Data Indexer:** after a cooperative study with HUTER partners in order to find out new requirements for data access tools in the HUTER context, it was detected the possibility of querying of genomic data. The technical solution proposed by Bahia Software to deal with issue is based on the caching genomic information in a database which is specialized in the treatment of large structured information such as Elastic Search.

6.3.1. Scalable Storage

To cover the functional requirements previously identified, storage component keeps on relying AWS S3 features about scalability, availability and information security, as well as data recovery. These features were detailed in previous documents so no further information is expected to be required since no additional needs have been arisen.

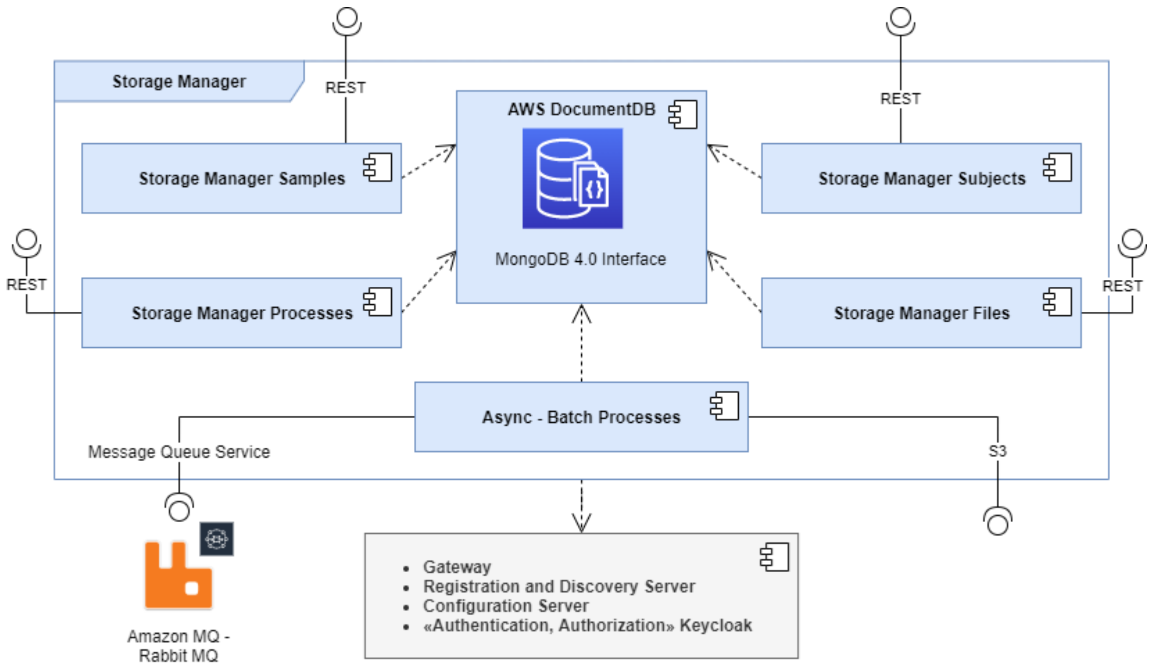
6.3.2. Storage Manager

This component provides a management core for the stored files and metadata. It offers the capability of indexing any files in the platform as well as tracking each one not only linking it to a subject but also through internal process like ingestion, analysis and so on. The indexation process also provides the possibility to discover files from common information what means a powerful tool for researchers' investigations.

The storage manager exposes an API REST of services deployed on the previously mentioned ECS service in order to decouple the data access layer. Following this approach, any authorized client in the SSO can access to the stored information regardless the database implementation which is hidden behind the API.

In this final version of the architecture design all the required components are shown in Picture 9 and detailed next:

- **Database:** for data and metadata indexation a NoSQL database has been chosen. The main reason for this decision is the flexibility of these databases to add new structured information avoiding the limitations of a relational model. In order to deploy this solution an AWS DocumentDB instance has been created because it provides a MongoDB 4.0 compatible interface. Such an interface is widely used in software development so there are several frameworks that help in its integration.
- **Storage Manager Subjects:** this service is aimed to give access to the indexed subject information regarding the HUTER Storage API requirements of security and reliability.



Picture 10 - Storage Manager

- **Storage Manager Samples:** like the Storage Manager Subjects service, this one provides the possibility to manage the sample data indexed in the NoSQL database through and API REST.
- **Storage Manager Processes:** as for HUTER Storage API requirements, this service provides access to Processes information. It is remarkable that it includes a method that uses an MQ service to manage asynchronous requests for file validation.
- **Storage Manager Files:** since this service is part of Storage API, it runs in the same way as the rest of services in the API but it is aimed to query stored files and their related metadata.
- **Async – Batch Processes:** this service is a technical solution to cope with long time execution processes that follows a batch approach for messages consumption. As for the architecture design, every ingestion request has to ask for the validation of the uploaded files by sending a message to a queue platform. Each queued message will be treated by a validation batch process, so the validation can be accelerated by increasing the number of instances so as to get rid of the queued messages.

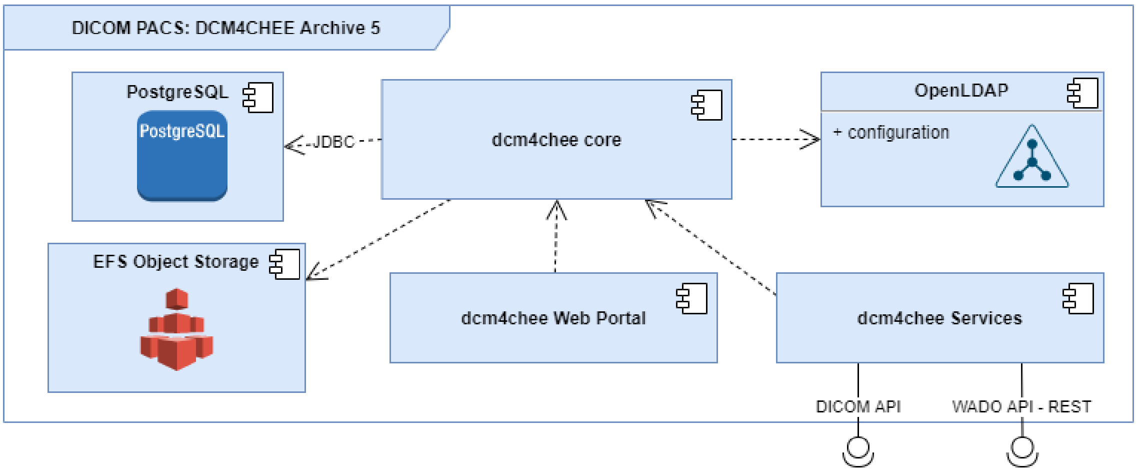
  As for its functionality, it is expected to be used just for the execution of file validations before indexing their metadata in the database. Albeit, this design allows HUTER Platform to redefine any other expensive task with a batch approach.

  - **Message Queue Service:** the queue platform has been deployed using the Amazon MQ Service that allows the definition of an Active MQ or Rabbit MQ cluster. The decision of using the Rabbit MQ implementation was the fact that this tool is widely used and means a well known reliability.

6.3.3. DICOM PACS

The DICOM proposal made by Bahia Software involves not just the transformation of research images into DICOM image format but also the deployment of an imaging server following the DICOM Communication standard.

In Picture 11, the complete architecture of a standalone server is shown and later briefly explained because no changes have been applied to this component since deliverable D2.1.



Picture 11 - DICOM Archive and Communication

A brief summary of DCM4CHEE PACS components is the next:

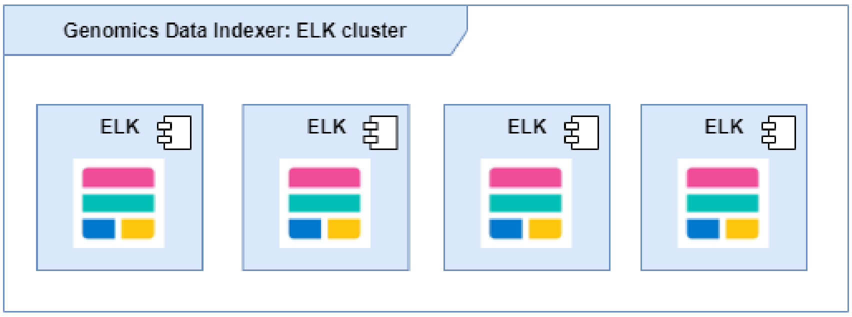
- **PostgreSQL Database:** this database is used to store the index of DICOM objects stored in the server.
- **EFS Object Store:** is a reserved file system aimed to keep the DICOM Objects indexed in the PACS.
- **OpenLDAP:** it is a widely used tool for managing credentials and configuration. In this proposal it will be used just for configuration because it will be isolated from the outside of the HUTER network.
- **dcm4chee-core:** it is an already existing component that involves all the logic to manage DICOM objects inside the PACS.
- **dcm4chee Web Portal:** it is a web interface that allows to manage the dcm4chee PACS as well as querying data and access to it.
- **dcm4chee Services**: is a layer of REST and DICOM directives that allows clients to access DICOM metadata and data using both methods defined in DICOM standard.

6.3.4. Genomics Data Indexer

This new component has been aimed to work as temporary storage for omics data obtained from genomic VCF files. Regarding to the nature of the genomic data what means large structured information, the deployment of a specialized tool in managing structured information has been proposed.

Bahia Software proposal is based on the experience of IT Services in dealing with similar issues for years such as the log data storage and its exploitation problem. Therefore, an effective solution is to use an ELK environment which consists on an ElasticSearch, Logstash and Kibana cluster. Using this approach, we are expected to provide a genomic indexer backend to query and compare genomic data on demand by a web based application.

However, according to the suggested use of this feature by the researchers, the indefinite storage of such amount of data will not be possible in terms of cost-efficiency. That is why it has also been proposed the use of this solution as an on-demand cache so as to execute studies over a clearly defined set of files.

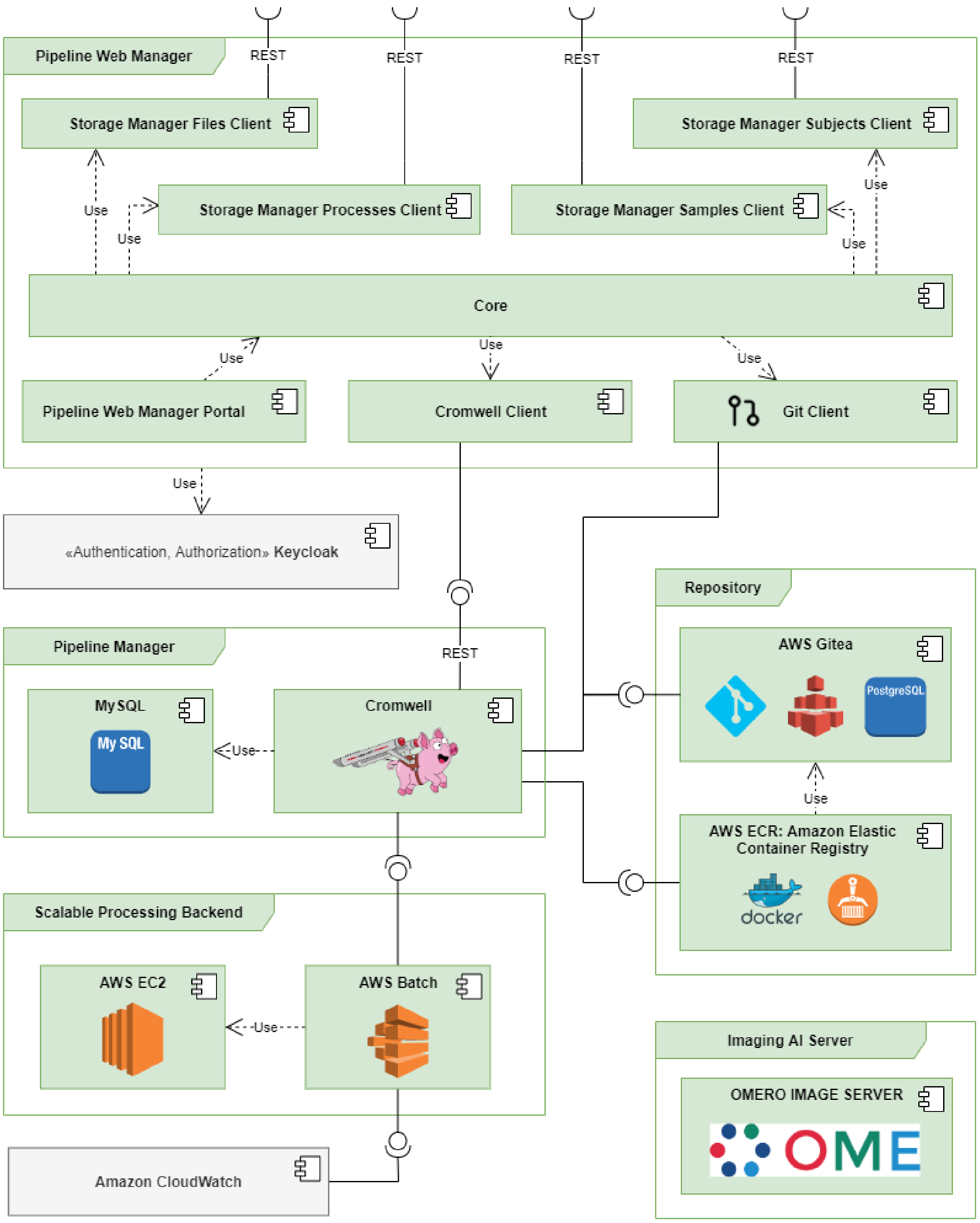


Picture 12 - Genomics Indexer proposed solution

6.4. Processing

Processing module should allow collaborators to run analytical tools over the ingested data. The result of this analysis will be automatically ingested into the platform linked to the input data in order to keep the information tracking.

In Picture 13, the complete current design of processing module. There are some additional features in this new proposal but the agreed features of the module remain such as Pipeline Web Manager, the SSO integration, information tracking and so on. Unlike the previous design version of the HUTER platform, in this proposal a new component has been included aimed to provide AI algorithms to be applied over DICOM images.



Picture 13 – Processing

In the final design of the module four main components are distinguished:

- **Pipeline Web Manager:** a web application to provide a user-friendly interface for running pipelines.
- **Repository:** it consists of a set of tools that allow the storage of software to be run in the platform either code versioned in a git-like repository or docker images in a container repository.

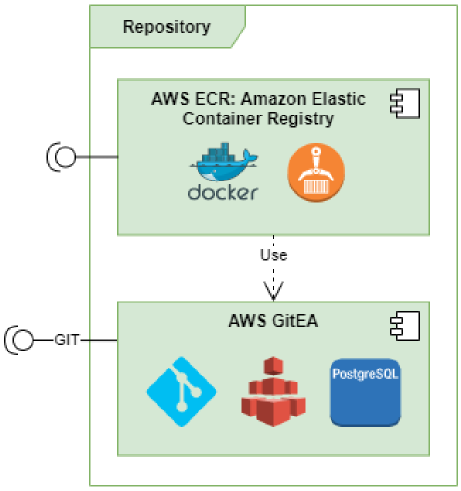
- **Pipeline execution:** this component has been drawn in Picture 13 – Processing as two separated technical components however, both of them work together to provide a flexible solution for pipeline execution over stored files. The scalable pipeline execution system will be provided deploying a Cromwell environment integrated to the storage module and the repositories for WDL development and docker images. The backend processing power is provided by a AWS Batch Services that allows the deployment of dockerized machines on demand for pipeline running.

  But not only analysis process can be run in this component but the dicomization process proposed by Bahia Software will be executed in it to get advantage of the tracking workflow developed.

- **Imaging AI Server:** the DICOM Viewer will be complemented with the possibility of applying AI algorithms over the viewed images. Regarding this proposal, an OMERO server will be deployed to run these algorithms through an integration with the Viewer.

6.4.1. Repositories

Regarding to collaborators needs, two repositories have been created to develop and create tools for data processing. Picture 14 demonstrates the definition of both of them and the relation between them.



Picture 14 - Repositories structure

A first repository for code development has been created deploying gitEA over an AWS EFS as file system and a postgreSQL as database. This repository is divided in two sections: WDL section and docker-compose section. Both are divided by collaborator's organization, too.

WDL section will contain pipeline definitions using WDL format compliant with HCA guidelines. This repository based on the GitEA provides the benefits of git repositories as well as OpenID Connect integration with Keycloak. As for docker-compose section, it will provide the capability of defining docker images including external or internal tools for data analysis.

Another repository to contain docker images is provided through AWS ECR Service. This self-contained repository definition is integrated with the docker-compose repository so as to automatically generate docker images on a push event on the git repository.

6.4.2. Pipeline manager + scalable processing backend

These two modules will be explained together because both are necessary to achieve data processing.

The Broad Institute's Cromwell is purpose-built for this need. It is a workflow execution engine for orchestrating command line and containerized tools. Most importantly, it is the engine that drives the GATK Best Practices genome analysis pipeline.

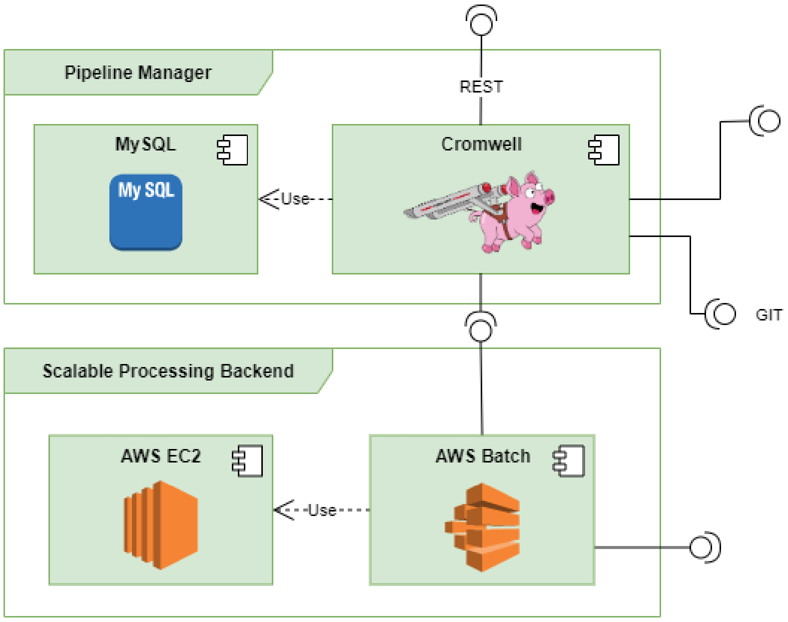
Workflows for Cromwell are defined using the Workflow Definition Language (WDL), a flexible meta-scripting language that allows researchers to focus on the pieces of their workflow that matters. That is the tools for each step and their respective inputs and outputs, and not the plumbing in between.

Genomics data is not small (on the order of TBs-PBs for one experiment), so processing it usually requires significant computing scale, like HPC clusters and cloud computing. Cromwell has previously enabled this with support for many backends such as Spark, and HPC frameworks like Sun GridEngine and SLURM.

Furthermore, a process to convert proprietary image formats from microscopes to DICOM files will be implemented with the aim of supporting DICOM transformation. This process must be integrated in the HUTER platform in order to be as traceable as the rest of the data processed in the platform. To accomplish this task, 4 proprietary formats will be studied and covered by the implementation of DICOM converters. Each converter must be designed according to a new DICOM proposal made by Bahia Software to NEMA consortium due to the lack of an existing one that covers the DICOM imaging format used by the HUTER partners.

In Picture 15 - Cromwell complete infrastructure, full infrastructure is shown taken into account that the deployed architecture (AWS) provides support for Cromwell, so AWS Batch has been configured as the backend provider to HUTER Cromwell instance.

Cromwell is such a flexible environment that will also be used to integrate the dicomization process proposed by BAHIA. As it was said, this should obey same rules of traceability as the rest of the data processed in the platform.



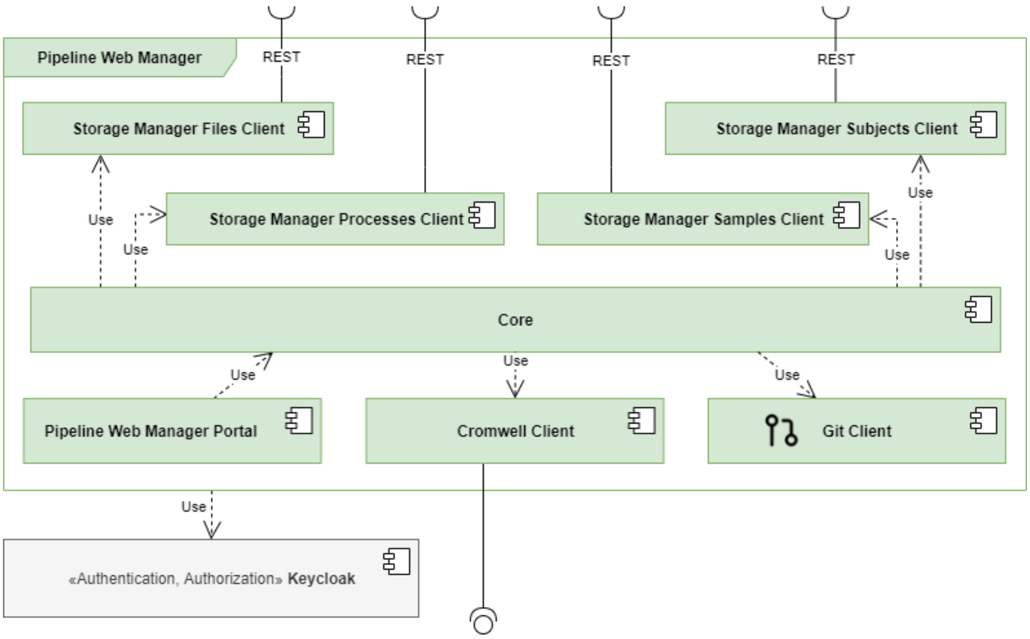
Picture 15 - Cromwell complete infrastructure

The most important features to point out are that:

- Cromwell instance runs over a MySQL database.
- Cromwell is also integrated with AWS ECR and git to retrieve both WDL and docker image reference to invoke the backend.
- Processing backend is supported by AWS Batch that provides a scalable system of AWS EC2 machines to run docker machines for pipeline execution. Every EC2 machine will be integrated with the AWS S3 service in order to access to input files and save the results.
- Cromwell exposes an API REST that allows users to request the execution of pipelines over stored data. However, as it will be seen in the next point, this will not be the designed way to invoke Cromwell.

6.4.3. Pipeline web manager

This application offers to the HUTER Platform users a web interface to interact with Cromwell in a user-friendly way. However, the main reason to provide this tool is not only to facilitate Cromwell executions but to ensure data tracking. The provided features allows the HUTER Platform users to access to a list of existing pipelines which had been stored in the repositories as well as files in the storage module. Thanks to that, it will be possible to avoid data reference mistakes and assure data correlation.



Picture 16 - Pipeline Web Manager

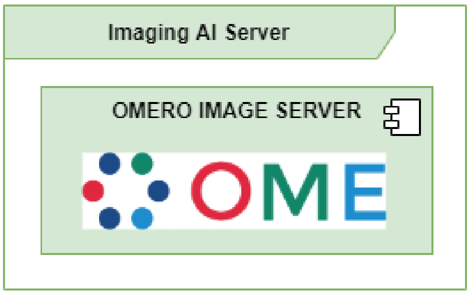
In Picture 16, components of Pipeline Web Manager are shown and next explained:

- **Core:** the main logic is contained in a core that orchestrates the consumption of the storage manager services to retrieve all the required data.
- **Pipeline Web Manager Portal:** it exposes a web interface integrated to HUTER SSO that applies a security layer. It also eases the access to data and metadata that can be analyzed.
- **Cromwell Client:** it is a REST client of the Cromwell Server to request the execution of a pipeline over preselected data.
- **Git client:** it means a client of the repository that allows the Pipeline Web Manager to retrieve the WDL definition from the git repository and send it to Cromwell through its client.
- **Storage Manager Files Client:** it is a REST client to query the files indexed in the platform through the Storage Manager. This service will provide the location of the files to be processed in the pipelines as well as the registration mechanism for the new ones.
- **Storage Manager Subject Client:** it is a REST client for subject metadata access.
- **Storage Manager Samples Client:** it is a REST client that allows the access to indexed sample metadata and allows users to select the sample metadata files to be used in the analysis process.
- **Storage Manager Processes Client:** it is a REST client that allows the pipeline web manager to provide the data tracking by registering each pipeline execution as a process in the platform as well as inputs and outputs.

6.4.4. Imaging AI Server

The Image AI Server was covered by the deployment of an OMERO Server. OMERO is a suite of tools for imaging analysis widely used in researching environments. It applies Artificial Intelligence algorithms on imaging data in order to detect some features in the images such as segmentation and region of interest detection.

The challenge of this integration is that the algorithms should be applied on DICOM images what involves managing the DICOM imaging peculiarities through an integration with de DICOM Viewer.

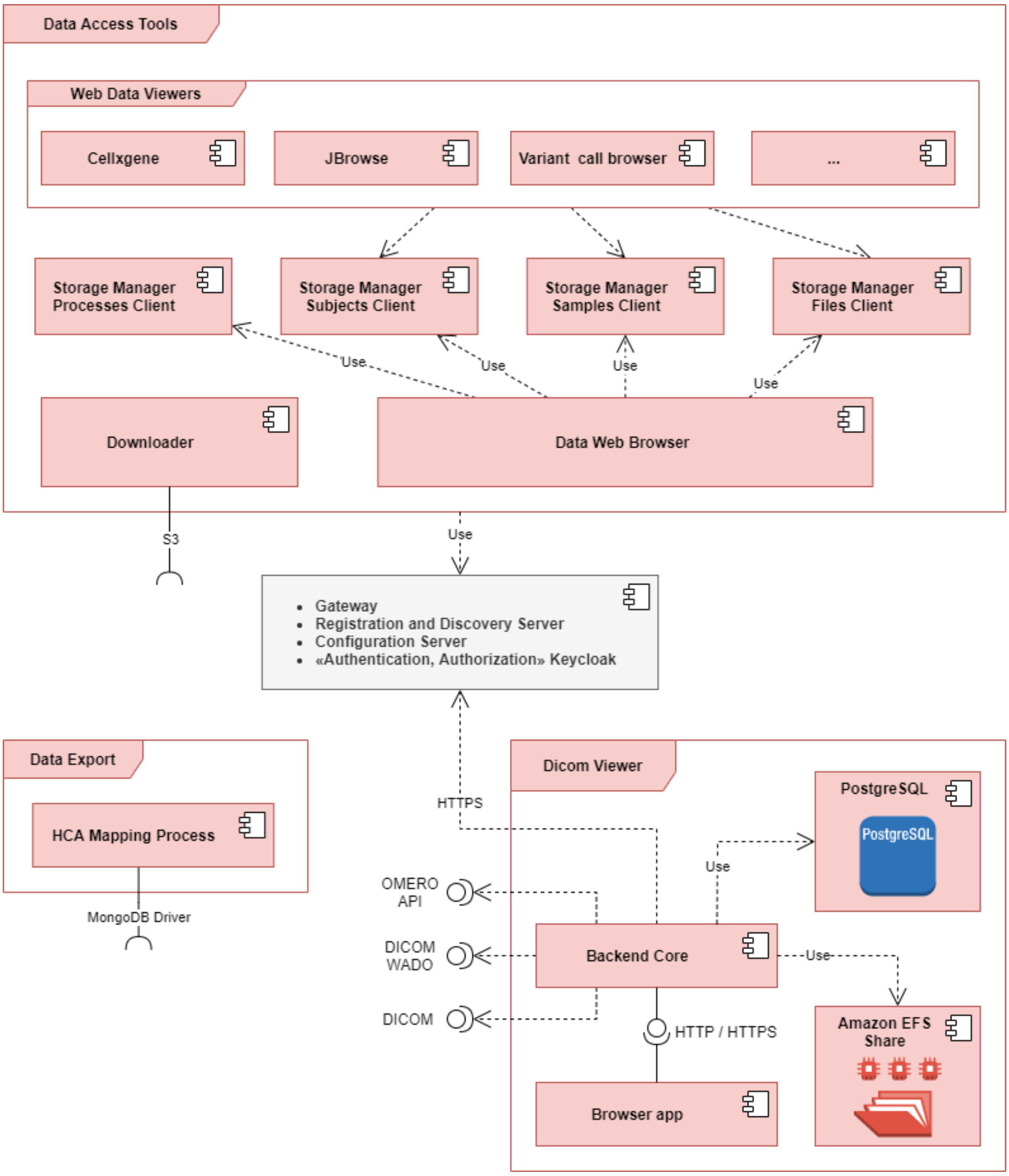


Picture 17 - Imaging AI Server

6.5. Data Access

Data Access module comprises a set of client applications and resources provided by the HUTER Platform to help users to visualize, access and interpret stored data according its nature. For instance, a tool for genomic visualization could be deployed as well as the mentioned DICOM Viewer. Both of them allow users to properly get access to the stored data adapting the presentation layer to the format of the data.

Picture 18 shows the general composition of this module which is a set of different applications that are proposed to be deployed and integrated into the platform and with its storage services. Further ahead a brief introduction of the tools will be done in order to clarify the whole final architecture. Albeit all these proposed data access tools are detailed in the document *HUTER_WP2_D2.3_Beta_version_of_data_access_tools*.

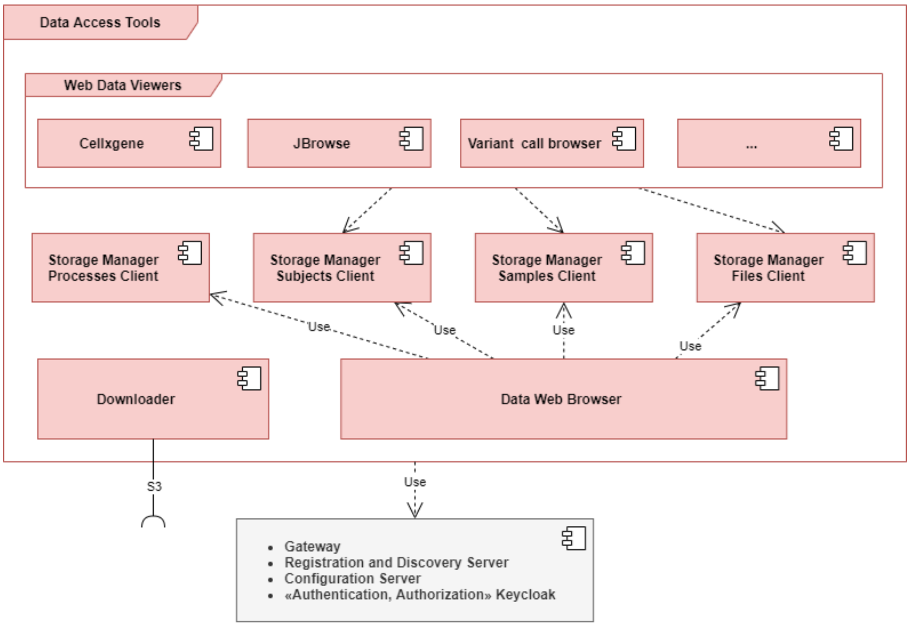


Picture 18 - Data Access module

6.5.1. Data Access Tools

The term Data Access Tools has been used to designate the set of applications that allows users to query, visualize or download stored raw data and metadata.

Before going on describing this set of tools, it is necessary to highlight the fact that is planned to deliver a document containing the detailed description of tools: *HUTER_WP2_D2.3_Beta_version_of_data_access_tools.* However, in order to deliver a complete architecture design, proposed solutions are shown in Picture 19 - Data Access Tools and described next:



Picture 19 - Data Access Tools

- **Web Data Viewers:** these are the proposed tools to complement the HUTER Platform functionalities regarding to the visualization of specific data formats. Due to the needed expertise to develop such tools it has been decided to deploy existing applications and their integration with the HUTER platform. Additionally, it is expected to invest a strong effort in the adaptation of some tools to a cloud and web environment.

  Next, the list of proposed tools is shown:

  - **Cellxgene:** this is an interactive data explorer for single-cell transcriptomics datasets in h5ad format. Initially, it was intended to be used on local machines but there are evidences of its deployment on web-based environments. So it is also expected to be deployed in the HUTER Platform as a web application integrated with the Storage Module by provisioning a dedicated server. The link to the project website is https://github.com/chanzuckerberg/cellxgene
  - **JBrowse 2:** it is a platform for visualizing and integrating genomic data. It runs as a web application that would be deployed performing the needed adaptations. The website of the projects is https://jbrowse.org/jb2
  - **Variant Call Browser:** according to likely needs of HUTER researchers, a custom tool for genomic variant querying is expected to be required. This tool means the web interface of the Genomics Data Indexer that will allow users to index the content of genomic files as well as the execution of queries and comparisons over these data. Due to the high volumetry of

this kind of data, its indexation in the Genomics Data Indexer has to be temporal in order to provide a balanced cost-efficiency yield.

- **Data web browser:** This component is a web application that allows users to query the stored metadata in the HUTER Platform. This portal does not provide direct access to the raw data, instead of that it offers access to lightweight data such as metadata and file index reference which can be used to download stored files. The download process will be done by generating a manifest file with the locations of the required files that should be passed to the Downloader tool.

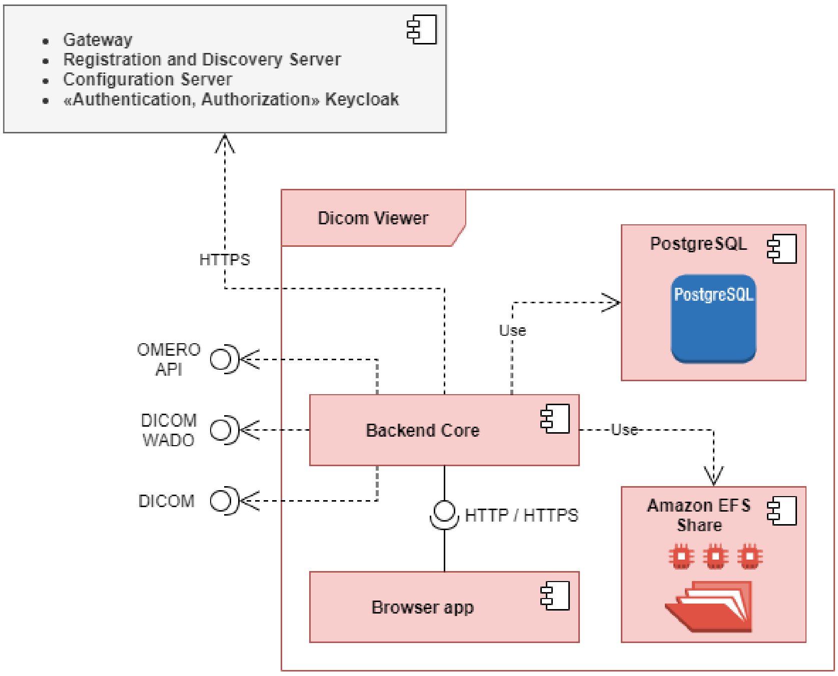
  As for accessing the indexed metadata, Data Web Browser will be integrated to HUTER Platform through the Storage Manager API. In order to consume the API services, it has to integrate clients for all managers that publish the information required by users. That means:

  - Storage Manager Subjects Client: for subject data querying.
  - Storage Manager Samples Client: for sample data querying.
  - Storage Manager Files Client: for file data querying and manifest file exportation.
  - Storage Manager Processes Client: for process data querying and data tracking.
- **Downloader:** this component is a tool for retrieving files from the storage module. It will be released as a command line interface application that will need the manifest file containing the locations of the desired files to accomplish the download. Moreover, this tool will assure that just authorized users can access to the files.

6.5.2. DICOM Viewer

DICOM Viewer is a necessary complement to the DICOM PACS. DICOM Viewer will consume the DICOM PACS API to access the images stored in it. Besides, it can interpret the DICOM Imaging Structure since DICOM Standard not only defines sections for metadata in its objects but also a variety of imaging meanings such as sequence of imaging in time as videos, sequence in space as TMA, pyramidal structures as pathologic imaging and so on.

Picture 20 shows the subcomponents and communication among them and the expected DICOM interfaces that it consumes for PACS communication. It has also been included the OMERO interface to achieve communication with this AI Analysis Server and the integration with the HUTER Platform SSO provided by Keycloak.



Picture 20 - DICOM Viewer subcomponents

6.5.3. Data Export

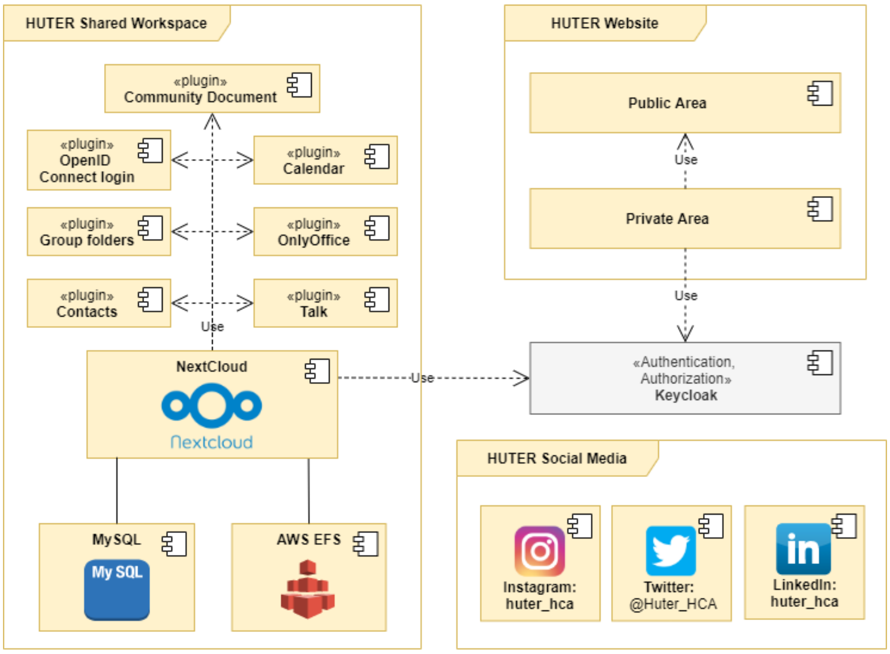
As for Data Export, it will be a tool for IT users. This means no collaborator unless BAHIA will need to use this tool because its aim is to help during the information transference from the HUTER Platform to HCA Data Portal.

In order to facilitate data analysis, the HUTER Platform will store data and metadata in the most suitable format for HUTER collaborators. However, it will always be taken into account that the content fits the HCA DCP guidelines.

6.6. Dissemination & Communication

The dissemination and communication module is composed of 3 independent elements. Each one covers complementary needs of the HUTER Project managers that were already mentioned in deliverable D2.1.

Next section reminds the components shown in Picture 21.



Picture 21 - Dissemination and communications components

6.6.1. HUTER Shared Workspace

This component offers to the team managers a shared space and tools for coordination tasks among all the HUTER partners and also for private team communication.

6.6.2. HUTER Website

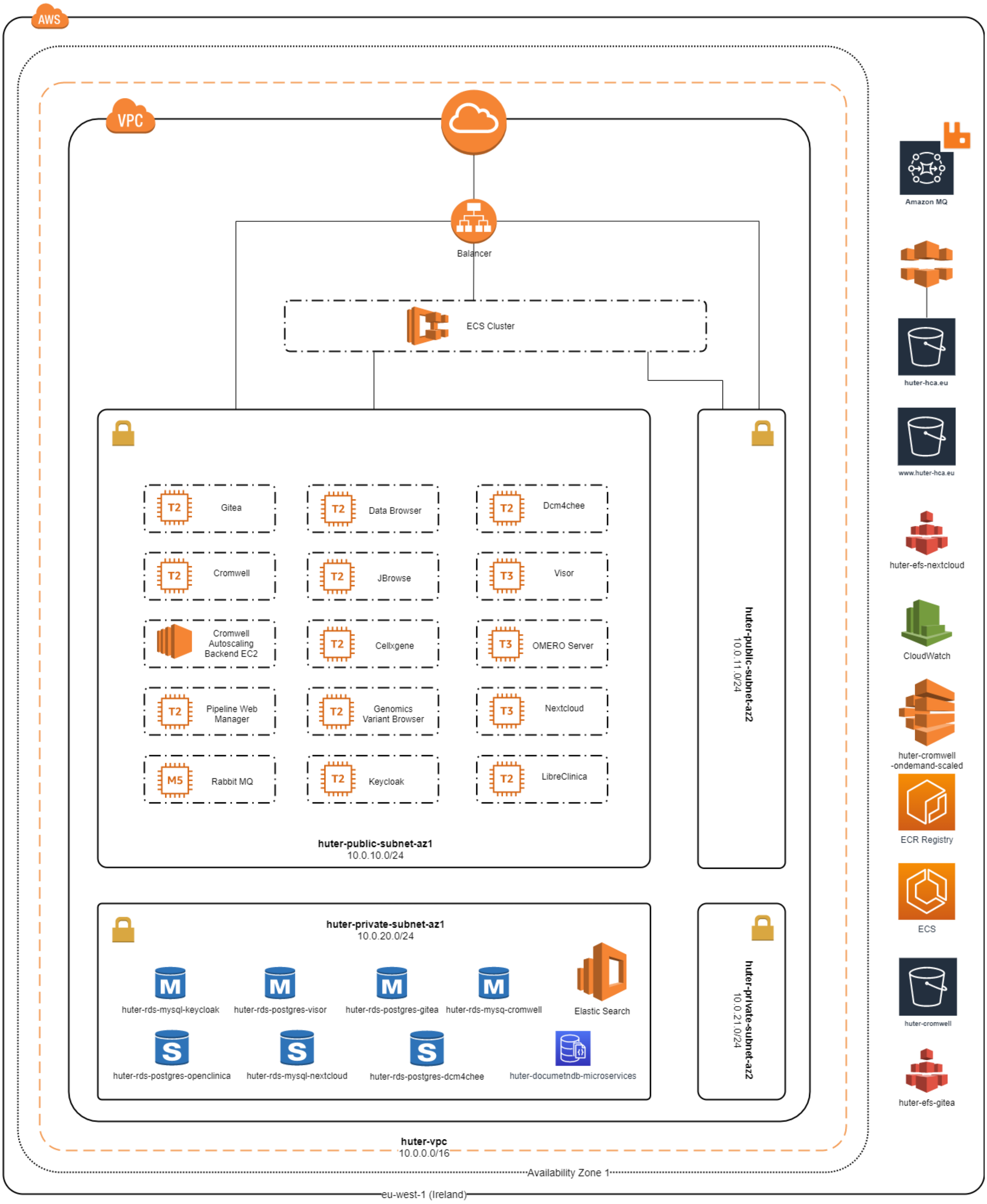
As part of the HUTER project requirements a website was created and in a regular basis update to publish the progress of the HUTER project. It also provides access for HUTER partners to a private area which works as an internal portal.

6.6.3. HUTER Social Media

Even when this is not a development module the creation of Instagram, Twitter and LinkedIn accounts is pointed out here because these social media platforms are powerful tools for communication objectives.

7. COMPONENTS DISTRIBUTION

In this section the final deployment architecture of the HUTER Platform is shown regarding to the current platform design. Some components of this architecture have already been introduced in the previous deliverable *D2.1* as well as the reasons taken into account in order to make the base infrastructure.



Picture 22 - Current deployed architecture

Due to AWS platform characteristics, main foundations of the HUTER platform have been maintained or even boosted. Just few solutions have to be designed and few others have been completely defined. As a consequence, some additional components have been deployed in order to support these solutions without damaging the toughness of the previous defined infrastructure.

So, just a brief description of the already defined deployment components will be explained in the current chapter and new ones will be detailed.

7.1. Deployment model

In Picture 22, the final infrastructure is shown as an overview of the complete set of components required by the HUTER Platform. We would like to highlight that all the components will be not deployed at the same time. Some of them will be deployed as planned and/or required by HUTER activities in order to optimize resources. The rest of the components are already deployed as reported in deliverable D2.2 and the new ones will be deployed on mentioned AWS services following the infrastructure shown in Picture 22 - Current deployed architecture.

In order to summarize the previous documented features in deliverable *D2.1*, main aspects of the deployment infrastructure will be listed next albeit new components will be highlighted.

7.1.1. Networking

A virtual private cloud (VPC) has been configured to isolate the HUTER Platform from the outside and add the first layer of the required security. Communication from the outside will be possible thanks to an external Gateway that allows incoming requests to reach HUTER services through an AWS API Gateway.

Under this VPC, 2 subnets have been configured and they are reachable from a Balancer located inside the VPC. Both subnets contain node machines where applications are running so this model works in an active-passive model. That is, if any node deployed on subnet 1 falls the balancer will redirect every request to the subnet 2. In *Picture 22 - Current deployed architecture* no nodes are drew under the subnet 2 in order to keep clarity about located components in each subnet illustrated in subnet 1. With regard to the subnets access, each subnet is isolated from the outside unless the access configured through the balancer and the communication channel to the databases if required. However, inside the subnet, all communications are allowed.

7.1.2. Application servers

Most of the current machines configured to support the deployed applications are instances of Amazon EC2 machines since these are the base unit for node definitions. Low performance instances of EC2 have been deployed so as to support the required functionalities but they can be upgraded as soon as required.

Besides, Amazon EC2 instances are flexible environments that allow for deploying almost any runtime for web applications or batch processes.

As for managing high-load processing, an AWS Batch service has been configured to auto-scale all the EC2 required power for heavy tasks in the platform such as omics analysis pipelines, imaging transformation into DICOM formats and so on. This service had been already introduced in deliverable D2.1 because it is a natural solution to work with the Cromwell instance that runs as pipeline Manager.

In deliverable D2.1, Elastic Kubernetes Service (EKS) had been introduced as the most suitable solution for microservice deployment. However, the AWS ECS Service has been chosen to replace it owing to its short complexity and good adaptation to the HUTER Platform requirements.

Regarding its benefits, AWS ECS is a foundational pillar for key Amazon services so it can natively integrate with other services such as, AWS Identity and Access Management (IAM) or Amazon CloudWatch providing node auto-scaling. Each node represents a service itself, released in docker container images that can be stored in the already existing ECR Service. ECS runs out of the subnets due to no backup is needed to support system fails. AWS platforms assured the ECS availability and, ECS assures the availability of every deployed service.

7.1.3. Databases

As it is shown in Picture 22, several instances of different relational and NoSQL database engines are required. Most of them have been expected in the previous deployment design but in this document some of them have been fully detailed or just appeared such as the ELK solution.

7.1.3.1. AWS RDS

To support relational database managers, AWS provides RDS service to set up, operate, and scale relational database in the cloud. It provides resizable capacity while automating time-consuming administration tasks such as hardware provisioning, database setup, patching and backups.

Amazon RDS is available on several database instance types as PostgreSQL 9.5 and MySQL 5 that were required for some HUTER tools.

7.1.3.2. NoSQL

As for data indexation in the HUTER Platform a noSQL was initially suggested. This idea remains as the best one that fits the project needs due to its flexibility and performance managing document-like information. If it is compared to a relational database, in the context of HUTER Project it would be necessary to defined lots of relational tables just to store few registries by table. On the other hand, a noSQL approach allows developers to focus on the functionality meanwhile the structure of the information is natively supported by the noSQL database. The unique concession has been the need of store data in JSON format what, as a matter of fact, is not a drawback. Other plus of noSQL databases is that have been demonstrated their scalability and performance in applications all around the world.

As for AWS support of noSQL database, it provides the DocumentDB service that offers as reliability as AWS RDS service for noSQL solutions. This service exposes a MongoDB 4.0-like access what makes it extremely interoperable and accessible by a well known interface.

7.1.3.3. ELK Cluster

The ELK stack is an acronym used to describe a stack that comprises of three popular open-source projects: Elasticsearch, Logstash, and Kibana. Often referred to as Elasticsearch, the ELK stack allows for aggregating structured data, analyze it, and create visualizations according to the user needs.

ELK is composed of 3 different solutions but the most suitable for HUTER purposes is the Elasticsearch which is an open-source, RESTful, distributed search and analytics engine built on Apache Lucene. The RESTful API can be exploited by the proposed Variant browser so as to perform queries and analysis over genomic data.

7.1.4. Storage

HUTER Platform will work with different types of storage systems depending on the characteristics of the data to be stored. The list of used services for storage was provided in deliverable D2.1 so in this section it will be briefly named:

7.1.4.1. Local machine filesystem

Some HUTER tools will need to access to labs local file systems in order to retrieve files for uploading. Even when those file systems are not part of the platform, HUTER tools must considers their conditions to properly work on them.

7.1.4.2. AWS EFS

Amazon Elastic File System (Amazon EFS) provides full managed elastic NFS file system to use with services deployed in the HUTER Platform. Some introduced cloud solutions need a local file system that will be provided through Amazon EFS.

7.1.4.3. AWS S3

Amazon S3 is an object storage built to store and retrieve any amount of data from anywhere on the Internet. It is a simple storage service that offers an infinitely scalable data storage infrastructure.

AWS S3 service is used to store raw data files in the HUTER Platform and it will be accessible through ad-hoc tools for data uploading and downloading. It will also be integrated with the processing module in order to make it possible to run pipelines on the stored data and save results again.

7.1.5. Out-of-the-box AWS Services

HUTER Platform takes advantage of ready-to-use AWS Services that provides required functionalities and tested integration methods.

7.1.5.1. AWS Batch

AWS Batch enables HUTER Platform users to easily and efficiently run hundreds of thousands of batch computing jobs on AWS. AWS Batch dynamically provisions the optimal quantity and type of compute resources (e.g., CPU or memory optimized instances) based on the volume and specific resource requirements of the batch jobs submitted. With AWS Batch, there is no need to install and manage batch computing software or server clusters that you use to run your jobs. AWS Batch plans, schedules, and executes your batch computing workloads on AWS EC2 machines.

AWS Batch was introduced in section 5.3 Processing as the backend for Cromwell executions that needs to run WDL over docker images.

7.1.5.2. Gitea

Gitea is a ready-to-run git solution that is secure, supported by AWS and easy to maintain. Gitea is a self-hosted Git service, somewhat similar to GitHub, Bitbucket and Gitlab. As well as support for Git revision control, it also provides issue tracking and development wiki pages. The system auto-updates itself with security fixes and is built in a transparent 100% open source process free of hidden backdoors. Gitea is a fork of Gogs, a lightweight code hosting solution written in Go and published under the MIT license.

Gitea runs in the HUTER Platform supported by a PostgreSQL database and an AWS EFS service. It is also integrated with IAM so a centralized user management can be done in the provided Keycloak.

7.1.5.3. AWS ECR

Amazon Elastic Container Registry (ECR) is a fully-managed Docker container registry that makes it easy to store, manage, and deploy Docker container images. Amazon ECR is used to make accessible all the docker images needed in the HUTER Platform. In an early deliverable D2.1, the very first approach was aimed to use ECR for storing the pipeline execution environment. However, with regard of the deployed Microservice platform, it is currently used to store docker images for services that form HUTER API.

7.1.5.1. AWS ECS

Amazon Elastic Container Service (Amazon ECS) is a fully managed container orchestration service. It is aimed reduce microservice administration tasks by a reliable and flexible solution.

As provided in deliverable *D2.1*, the very first approach was aimed to use EKS as Microservice platform, however, this ECS fits better HUTER infrastructure requirements avoiding new complexity.

7.1.6. Management tools

Finally, in this last section, it will be explained the management tools that offer an overview of the platform health. All of them are Out-of-the-box AWS Services but they are independently described because of their functionality.

7.1.6.1. Amazon CloudWatch

Amazon CloudWatch is a monitoring and observability service that provides data and actionable insights to monitor applications, respond to system-wide performance changes, optimize resource utilization, and get a unified view of operational health.

7.2. Technological environment

The technological environment regarding to the deployment platform has suffered from no changes so this section remains the same as the one proposed in deliverable D2.1. No further features have been addressed because AWS Platform has offered a complete deployment solution providing a full management set of tools.

8. INTEGRATION WITH EXTERNAL APPLICATIONS

HUTER Platform does not need external applications to provide their services. HUTER Platform will expose some entrypoints API-like to gather and release information during the project execution. However, as a final step, a data migration must be done towards HCA Platform.

8.1. HUTER API

HUTER Platform will expose some entrypoint to accept new data upload and data querying. Regarding data processing, it will be managed by HUTER applications with web interface so direct data access for processing will never be available.

In this early step, no detailed API definition has been performed yet. However, the API purposes will be related to these issues.

8.1.1. Data ingestion: API for data submission.

This API must provide methods:

- To validate user access.
- To create or delete a data submission packages related to a sample.
- To add, modify or delete files in the data submission package.

- To request uploaded data consolidation in the storage module.
- To query submission package status.
- To gather and transform subject and sample data in order to be indexed as HUTER metadata.

8.1.2. Data querying

This API will provide methods to:

- Query metadata applying filters related to subject, sample, file or process metadata.
- Download data.

8.1.3. DICOM API

DICOM API is provided by the implementation of the DICOM PACS DCM4CHEE Archive 5. Through a REST API known as WADO, any DICOM client can get communicated to the PACS so as to store, query or retrieve any DICOM data.

Current DICOM API will be consumed by the DICOM Viewer to provide its functionalities but it could also be exploited by any DICOM client.

8.2. HCA alignment

After attending two meetings with HCA-DCP leaders, the foundation of the guidelines to assure the compatibility of the HUTER platform with the HCA-DCP has been established. As a result of these conversations, the full alignment with the HCA-DCP has been defined through the answering next issues:

- Communication protocol: HCA-DCP group is working on the development of IT services that reduces the human effort for metadata ingestion. However, meanwhile these services would not be published, the unique way for metadata communication is the generation of spreadsheets.

  In order to deal with this blurred requirement, HUTER Platform takes into account a future tools to translate the HUTER data model to the HCA available one at the moment of the project life end.

- Ontology: current HCA ontologies are not ready to support some of the variables related to the HUTER Project. However, HCA-CP team has offer to HUTER partners some early solutions such as:
  - Provide no-supported data in additional files linked to the main metadata.
  - Work together with HUTER partners to improve their ontologies in order to support the field values managed in HUTER studies.

As far as IT supporters are related, both issues are easily manageable from a technical sight and they have been taken into account to plan a final effort for building the Data Export tool.

9. STANDARDS, DESIGN RULES AND STRUCTURE

Due to the nature of the managed information in the HUTER Platform, it will be compliant with the Ethical Plan defined to ensure subject's rights. So de-identification and data access security are essential features that belong to the HUTER Platform basements.

According to that goal, during the development of HUTER tools some global and de facto standards will be included in the design in order to get a secure and reliable platform. Implementing software standards also helps to leverage the flexibility of the HUTER Platform so as to easily integrate it with HCA and third-party clients, even when this last option is not expected.

Furthermore, the integration of the DICOM standard in the platform to provide a transformation process and a visualization tool in this format can be a starting point for the research and clinical environments. In that way, DICOM offers to the research world a unified image format and communication standard consolidated and proved in clinical environments. Besides, this format includes related subject information in well-known ontologies that can be boosted by the researcher's contributions.

9.1. Technologies

The final stack for software solution development has not changed from the suggested one in deliverable D2.1.

9.2. Methodologies

There are also no changes about the working methodology.