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:	30/09/2020
Submission Date	02/10/2020
Leader of this report:	BAHIA
Deliverable no:	2.1
Deliverable name:	Platform architecture design, interconnectivity aspects and standards to use, fully aligned with the HCA guidelines
Dissemination level:	**Public**

Version History

Version	Date	Details
1.0	30/09/2020	First 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 2.1

Platform architecture design

Table of Contents

Table of Contents 2

List of Acronyms 6

1. PURPOSE OF THIS DOCUMENT 8

1.1. Related documents 8

2. EXECUTIVE SUMMARY 9

3. INTRODUCTION 10

3.1. HUTER project 10

3.2. Requirements 11

3.2.1. Functional requirements 12

3.2.2. Platform transverse requirements 15

4. PLATFORM ARCHITECTURE 17

4.1. Data Ingest 18

4.2. Storage 19

4.3. Processing 19

4.4. Data Access 20

4.5. Security, Management and Support Layer 21

4.6. Dissemination & Communication 22

5. SUBSYSTEMS DESIGN DESCRIPTION 23

5.1. Security, Management and Support Layer 23

5.1.1. Indentity and Access Management 24

5.1.2. Log Management 25

5.1.3. Monitoring 25

5.1.4. Gateway 25

5.1.5. Scalable services: Gateway, Registration & Discovery Server and Configuration Server 26
5.2. Data Ingest 26
5.2.1. eCRF Manager 27
5.2.2. Ingestion Manager 29
5.2.3. Broker – Uploader 30
5.3. Storage 31
5.3.1. Scalable Storage 32
5.3.2. Storage Manager 33
5.3.3. DICOM PACS 34
5.4. Processing 35
5.4.1. Repositories 36
5.4.2. Pipeline manager + scalable processing backend 37
5.4.3. Pipeline web manager 38
5.5. Data Access 39
5.5.1. Data Access Tools 40
5.5.2. Storage Service Client 41
5.5.3. Data web browser 41
5.5.4. DICOM Viewer 41
5.6. Data Export 42
5.7. Dissemination & Communication 43
5.7.1. HUTER Shared Workspace 43
5.7.2. HUTER Website 44
5.7.3. HUTER Social Media 45
6. COMPONENTS DISTRIBUTION 46
6.1. Deployment model 46
6.1.1. Networking 48
6.1.2. Application servers 48
6.1.3. Databases 49

6.1.4. Storage 49
6.1.5. Out-of-the-box AWS Services 50
6.1.6. Management tools 51
6.2. Operational model 52
6.2.1. Management deployment model 53
6.2.2. Ingestion deployment model 54
6.2.3. Storage deployment model 55
6.2.4. Processing module deployment model 56
6.2.5. Data Access module deployment model 57
6.2.6. Dissemination & Communication module deployment model 58
6.3. Technological environment 58
6.3.1. Security and access control procedures 59
6.3.2. Continuity, capacity y availability 60
6.3.3. Operational and system administration procedures 61
7. INTEGRATION WITH EXTERNAL APPLICATIONS 62
7.1. API 62
7.1.1. Data ingestion: API for data submission 62
7.1.2. Data querying 62
7.1.3. DICOM API 62
7.2. HCA alignment 63
8. STANDARDS, DESIGN RULES AND STRUCTURE 64
8.1. General considerations about development. Technology justification. 64
8.2. Technologies 66
8.2.1. HUTER Platform 67
8.2.2. DICOM Viewer 68
8.2.3. LibreClinica 69
8.2.4. Dicomization process 70
8.2.5. Web 70

8.2.6. Deployment environment 70

8.3. Methodologies..... 70

List of Acronyms

API	Application programming interface		**AWS**	Amazon web services		**BAHIA**	Bahía Software S.L.U.
BAM	Binary Alignment Map format		**BCL**	Binary base call format		**DEV**	Development
DICOM	Digital Imaging and Communication On Medicine		**eCRF**	Electronic Case Report Form		**EFS**	Elastic file system
EML	Ecological Metadata Language		**GBs**	Gygabytes		**GDPR**	General Data Protection Regulation
GNU	General public license		**HCA**	Human cell atlas		**HCA-DCP**	Human Cell Atlas Data Coordination platform
HPC	High-performance computing		**HTTPS**	Hypertext Transfer Protocol Secure		**HUTER**	Human Uterus Cell Atlas
IAM	Identity and Access Management		**IT**	Information technologies		**JDBC**	Java Database Connectivity
JSON	JavaScript Object Notation		**JWT**	JSON Web Token		**LDAP**	Lightweight Directory Access Protocol
MVC	Model–view–controller		**MVVM**	Modelview-view-model		**PACS**	Picture Archiving and Communication System
PBs	Petabytes		**PMI**	Project Management Institute		**PRD**	Production
RDS	Relational Database Service		**REST**	Representational state transfer		**S3**	Simple Storage Service
SOAP	Simple Object Access Protocol		**SSH**	Secure Shell protocol		**SSL**	Secure Socket Layer
SSO	Single Sign-On		**TBD**	To be determinated		**TBs**	Terabytes
TLS	Transport Layer Security		**UML**	Unified Modeling Language		**VPC**	Virtual private cloud

WDL	Workflow description language		**WP**	Work package			

1. PURPOSE OF THIS DOCUMENT

The purpose of this report is to present the first version of the HUTER Platform architecture design, interconnectivity aspects and standards to use, fully aligned with the HCA guidelines, being one of the key deliverables of WP2 defined in the Grant Agreement.

In order to adapt the design to refined requirements in view of progress in raw data collection, metadata definition, agreements on more standard high-throughput pre-processing methods and implementation of advanced analytical methods (WP4-WP6), as well as the final features of the enabling cloud-based infrastructures (WP1) and the on-going evolution of HCA guidelines and definition of common protocols and standards, it is foresee a deliverable entitled "*WP7_D7.1_Final_design_of_HUTER_platform_architecture"* with due date month 12. Such document will be an update of this document after first year of the project as part of the WP7: Platform integration.

This document has been developed by BAHIA as lead of the WP2: Platform infrastructure, in close collaboration with HUTER partners.

1.1. Related documents

Documents linked to current actions to be delivered:

HUTER_WP2_D2.2_Deployment_of_HUTER_cloud_infrastructure

Documents linked to future actions to be delivered:

HUTER_WP2_D2.3_Beta_version_of_data_access_tools

HUTER_WP2_D2.4_Final_implementation_of_data_access_tools_and_DICOM_visualization_tool

HUTER_WP7_D7.1_Final_design_ of_HUTER_platform_architecture

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 of cell sequencing and advanced microscopes data but also functionalities to satisfy the specific requirements of HUTER researchers in terms of collaborative work and advanced tools. This document develops the first version of the HUTER platform architecture design, aligned with the HCA guidelines as well as with the requirements of the HUTER project researchers.

The first version of platform design 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-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.

In base of these requirements it has been proposed 6 general components to construct the architecture system with their interactions: Data Ingest, Storage, Processing, Data Access, Security, management and support layer and Dissemination & Communication components. These functional modules are aligned to HCA-DCP platform design, but they address functionalities in a more direct way to deliver a functional HUTER Platform quickly. Data ingest provides the very first communication tools with the HUTER Platform, checking data quality and integrity as well as initiating information track. Next module is related to storage that ensures data indexation and availability through a service layer that relies on AWS infrastructure. In regards of Data Access, it is a set of tools that allow data querying and downloading for authorized users. This authorization functionality is provided by the Security, Management and Support layer that also includes services and tools to endorse system availability, capacity and monitoring. Finally, an independent module to help HUTER users in Communication and Dissemination tasks is composed of an online workspace and publishing tools like a website and social media accounts.

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, omics and other data formats are expected) and collaborators laboratory workflows. Furthermore, some first approaches related to the integration with external applications were included such as an early definition of APIs functionalities. However, this information as well as the beta access tools are object of later deliverables.

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 it 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 line, 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 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 and to comply with HCA sharing rules is to transfer gradually the data 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).

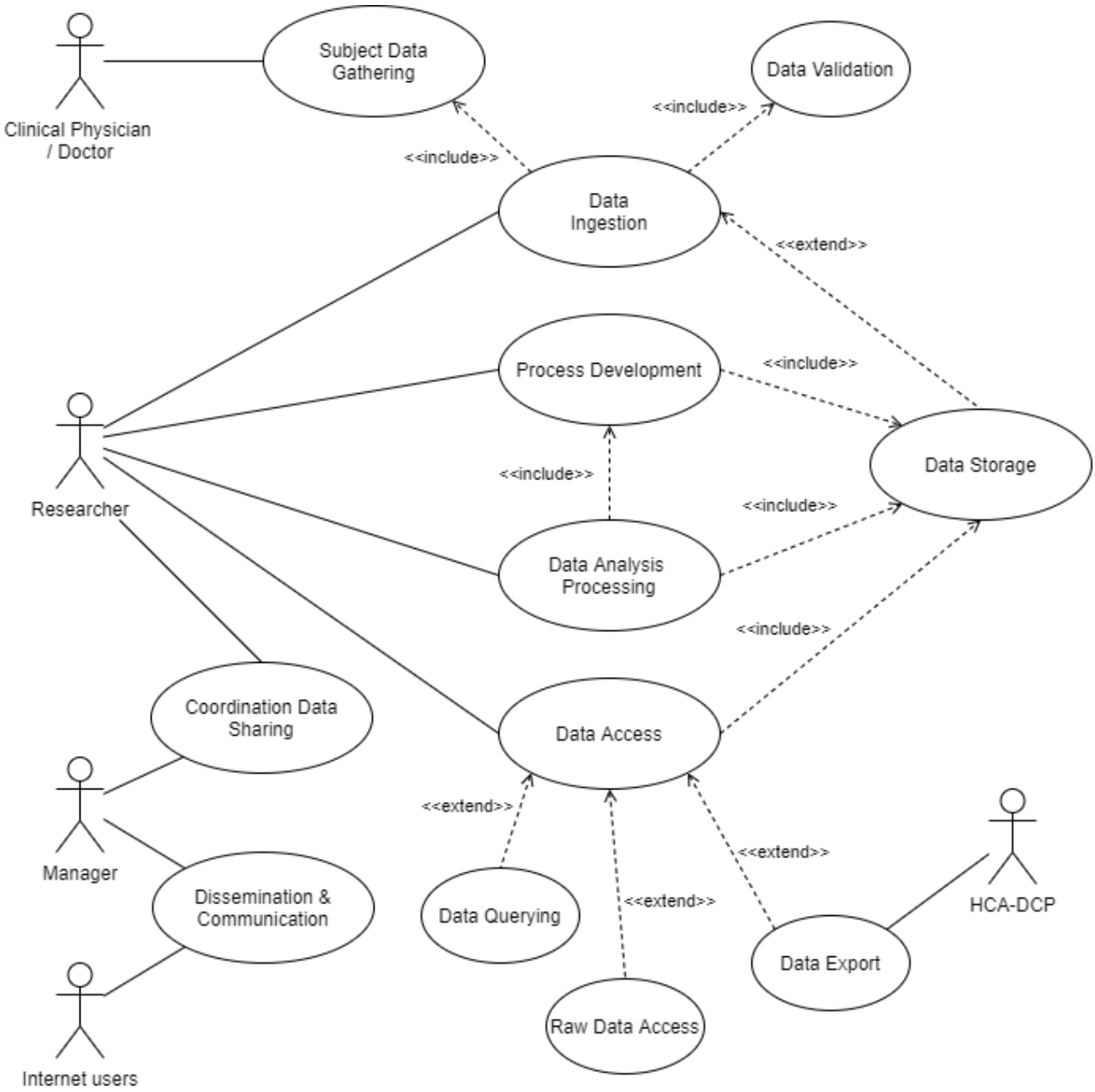
This deliverable 2.1 defines the first version of the HUTER platform design, fully aligned with the latest advancements of HCA-DCP. In order to adapt the design to refined requirements in view of progress in raw data collection, metadata definition, agreements on more standard high-throughput pre-processing methods and implementation of advanced analytical methods (WP4-WP6), as well as the final features of the enabling cloud-based infrastructures (WP1) and the on-going evolution of HCA guidelines and definition of common protocols and standards, it is foresee a deliverable entitled *HUTER_D7.1_Final_design_of_HUTER_platform_architecture* with due date month 12 in the context of WP7.

3.2. Requirements

One of the objectives of the HUTER platform is to support HUTER professionals in their hard research activities providing not only advanced processing and hosting capacities until the HCA-DCP platform is ready to host their different kind of data, but also connect and share data between HUTER researchers located in different countries to enhance their collaborative efforts. In order to accomplish these objectives, the main requirements of the HUTER platform were gathered from different HUTER partners to design an open-cloud system that will be accessible for any partner independently of their location. Additionally, HCA guidelines were taken into account with the aim to keep data compatibility with HCA-DCP, which will be the final repository of the data after the project ends.

In this line, the current HCA-DCP design guidelines were taken into account to design the HUTER platform, in so far as requirements between platforms were comparable. Therefore, current design of HUTER systems tends to cover a subset of HCA Platform functionality that fits collaborators needs, pruning unnecessary features under HUTER project context and adding new others with a view to accelerating and facilitating researcher teams work. A context diagram using UML rules shows the main functionalities required by the HUTER partners as well as the main target users of each functionality (Picture 1). These capabilities are focus on the efficient integration and execution of the different steps in a study, from data gathering to data processing. In the following section, main aspects of functionalities are explained and how they can be performed by users. Furthermore, it is also important to highlight that transversal requirements regarding security, integrity and anonymity of data are implemented following Ethical and Data management plans fully explained in deliverable documents *HUTER_WP1_D1.1_Ethics_Plan* and *HUTER_WP9_D9.2_Data_Management_Plan.*

3.2.1. Functional requirements



Picture 1- HUTER Platform context diagram

- **Subject Data Gathering:** This feature is required by HUTER partners to allow data collectors to register and manage electronically all the clinical information regarding subjects and samples under a defined protocol.

  As data collection will be performed by HUTER partners from different European countries, the software solution must be accessible from any location. Additionally, data registration will be carried

out by different kind of professionals who may not be tied to scientific environments therefore different user profiles must be managed to guarantee secure data registration in the platform.

Another important point is that the solution must provide the ability to configure different datasets to be collected. Since two types of samples are going to be collected during de project (endometrial biopsies and whole uterus samples), HUTER researchers will require specific information based on the sample type and analysis process.

Regarding secure data registration, it must be covered by validation processes that ensure data reliability and a curation process should also be included in order to correct mistakes or standardize data. This is the reason why subject data gathering is a critical step to get a study subject group that fits the protocols of Ethical and data management plans.

The underlying technology of this functionality must obey every ethical requirement detailed in Ethical and Data Management Plan as well as the rest of functionalities.

- **Data Ingestion:** HUTER researchers must be able to store data of samples as well as subject information into a common data storage system. Data ingestion also has to offer the first data indexing step in the platform so as to correlate subject data with sample data files from other sources.

  In a first approach, because several different file formats of advanced microscopy images and cell sequencing techniques are expected to be uploaded (e.g. SVS, FASTQ, BAM, BCL files among others), the platform must support and accept a variety of different file formats. Furthermore, it is important to highlight not only the number of file formats involved but also the big size of these files generated from research laboratory source systems such as advanced microscopes, state-of-art cell sequencers and other advanced laboratory equipment. Thus, this will require an efficient management of files regardless of their volume.
- **Data Validation:** Data Validation will provide a security functionality as part of Data Ingestion. It will assure not only prevention from unauthorized file formats but also quality and integrity of data submitted.
- **Data Storage:** The wide variety of European countries where partners are located makes common data storage as essential feature to keep information safe. Obviously, this resource must be accessible from anywhere in order to allow partners to share data of samples regardless of the study subject location. In this line, the HUTER Project requires this storage system to be scalable, secure and as fast as possible. Furthermore, it is required not only storage space but also data organization and traceability of source of samples (subject).

  Regarding data volume estimation, we are not yet able to provide accurate numbers or a complete list of data files involved due to the current initial stage of the project and the recent development

of the data source systems involved (e.g. recent cell sequencing technologies). However, very large files in the range of dozens of gigabytes that will come from advanced microscopes and cell sequencing equipment are expected to be uploaded. In this line, imaging and molecular data files such as FASTQ or BCL among others as well as data generated as consequence of their processing must be stored in the platform.

Furthermore, this feature must comply with the Ethical and Data Management Plan parameters as well as data integrity and availability.

- **Process Development:** Due to the state-of-art research to be carried out in the HUTER project, researchers and bioinformaticians may need to modify or even create new custom analysis protocols to apply to complex data generated from different sources. In this line, the HUTER platform must allow users to define new software tools even supporting not only auto compilation but also publication which is a feature required by HCA to comply with the collaboration agreement between both initiatives (HCA and HUTER).
- **Data Analysis Processing:** Another important requirement of HUTER project is to have the ability to perform standard or even custom process or pipelines over data generated from data sources (e.g. cell sequencing equipment) with the aim of obtaining relevant results through high-throughput analysis of complex cellular data. Therefore, the platform must provide a high-throughput processing area that allows researchers to execute their protocols. This processing capability must be expandable on demand according to research requirements due to the complex operations to be performed over large and complex data files.
- **Data Access:** The personal and sensible nature of data involved in HUTER research promotes that the stored data must be accessible guaranteeing authorization, subject anonymity and disponibility. Furthermore, indexing and searching tools are desirables with the aim of easing data retrieve.
- **Data Querying:** Data access should be efficient so data filtering must be a capability of the HUTER Platform.
- **Raw Data Access:** As essential part of data access, platform should provide mechanisms to retrieve stored files in original format.
- **Data Export:** Another of the main requirements that the platform must fulfill is to be able to act as a bridge between the data generated in the HUTER project and the HCA data platform which will be the main repository of the data after the project HUTER ends. In this line, the HUTER platform must be able to facilitate the ability to transform specific HUTER data formats into the HCA-DCP expected formats. For this reason, HCA-DCP design guidelines have been considered with the aim of synchronize data properly between both platforms.

- **Coordination Data Sharing:** HUTER platform must provide tools not only for complex research data handling but also for improving internal information flows among the partners, fostering information and ideas exchange as well as file and document sharing related to management of the project. These tools should have a major impact on the management of the project avoiding issues related to distance between partners that are located in different countries of European territory.
- **Dissemination & Communication:** HUTER consortium is committed to performing communication, dissemination and exploitation activities as described in WP8 and stated in the Grant Agreement signed. In this line, the platform must provide tools that enhance the diffusion of the milestones and achievements performed during the HUTER project as well as other relevant information such as project description, teams involved, news, etc. that must be publicly shared with the society.

Additionally, we want to highlight the research line included in the HUTER project proposal that aims to transform advanced images generated in biomedical research environments into the open medical image standard format DICOM. Thus, the platform will provide support to proprietary image formats that will be generated during HUTER project, but it will also have the functionality of transforming all these images to open medical image standard format, DICOM. This functionality aims to provide all HUTER partners with a unique format to access and visualize studies that generate medical imaging, to promote the application of such format in this research context and even improve the standard DICOM to facilitate the support of these images.

3.2.2. Platform transverse requirements

In addition to all these functional requirements identified in close collaboration with HUTER partners, a variety of transversal functionalities and features will be implemented that will underlie the rest of the layers. These necessary technical requirements to comply with Ethics and Data Management Plan gathered in deliverables 1.1 and 9.2 will be detailed below.

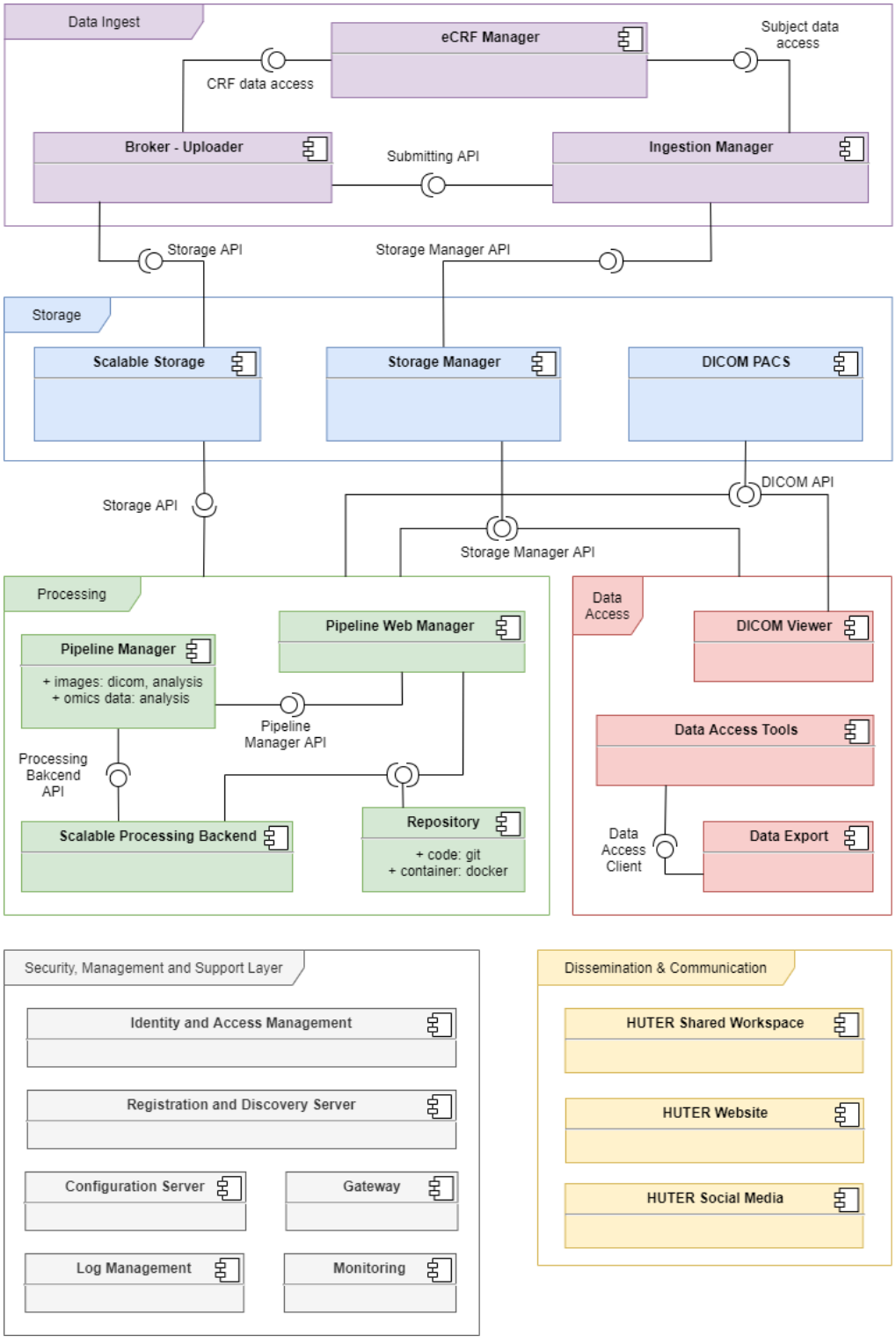
- **Anonymous traceability of the information:** The HUTER platform must provide the ability to relate samples information to its origin subject keeping the subject identification anonymized. This means that new information could be added to a subject profile so that it would be possible to keep track of any sample of each subject facilitating analytical tasks which involve different samples from a same subject. In addition to traceability features, high levels of anonymization are required to keep safe the identity of the subjects involved in the HUTER project. The clinical and molecular data as well as the processing results must be identified through virtually created identifiers following the Data Management Plan guidelines (Deliverable 9.2) so that the identity of the subjects will be concealed. Thus, personal identification cannot be obtained from the stored files or metadata. Both features, traceability, and anonymization, must be combined and supported by each HUTER platform

component. Thanks to this virtual identifier, every single file can be tied to their virtual subject code keeping traceability and avoiding the storage of real identification data.

- **Data Security:** information security is a wide term that comprises several aspects related to data reliability. In order to assure data security, the platform must provide tools and mechanisms that prevent from intentional or accidental destruction, modification, exposure or misuse of data. Although this feature is closely linked to data storing requirements, data security measures must also be taken for data transfer and processing.
- **Access Control: a**nother important transversal feature required for the HUTER platform must be the control of access not only to data but also to the different platform components. In fact, this functionality is the first step in data security as unauthorized users would not be able to view or modify data in the platform. Therefore, this feature will allow that only authorized HUTER users will be able to access to the different components of the platform as well as data. For this reason, the access control must be integrated into all components of the platform and must fit user requirements of each tool. In conclusion, a grained permission control must be managed to achieve this goal.
- **Location Independent Access:** HUTER project is a collaborative effort between 6 European institutions located in different countries which requires that access to the platform was possible from anywhere HUTER users were.

4. PLATFORM ARCHITECTURE

An overview of main components of the HUTER Platform architecture will be shown in this section. The subcomponents definitions will be included in section 5 SUBSYSTEMS DESIGN DESCRIPTION with the aim of clarifying both technical and functional reasons that support the decision of each selection.



Picture 2 - High level architecture diagram

A high-level draft of HUTER Architecture (Picture 2) shows functional modules required as well as the main technological components that will be provided. The aim of the schema is to link previous functional diagrams to further ahead information regarding the details of the architecture and components. Thus, this draft includes the technical solutions with the aim of locating the data entry points and the interfaces of the HUTER platform in a simple and understandable map.

The HUTER platform architecture design keeps an analogous structure to HCA-DCP architecture design. The reason of mimicking its skeleton is to face same problems with known solutions, focusing efforts on overcoming the specific challenges related to HUTER context. Thus, some features have been designed using more direct approach owing to the fact that HUTER project has a closer horizon than the HCA-DCP.

From a technical perspective, this design allows the HUTER platform to scale resources as required due to the fact that components have been defined seeking a low coupling and high integration. Furthermore, another remarkable feature of this architecture design is that allows the HUTER Platform to be deployed into a cloud environment. A cloud deployment will not only allow platform to fulfill the requirements that have been introduced in section 3.2 Requirements but will also provide additional benefits related to the system availability and scalability.

The main components will be listed and fully explained below. These will be grouped by their functional module relation in order to be properly linked to their functional requirements supported.

4.1. Data Ingest

In this module, data gathering functionalities are assembled to point out the interaction with sample data collection. Therefore, the HUTER platform must provide data registration software to savely collect, validate and store information of samples.

This module cannot be limited by a fixed set of sample and information formats since different types of sample will be processed. Furthermore, it has to provide tools to verify the quality and suitability of the data.

Based on these peculiarities, three components have been designed to accomplish these objectives:

- **eCRF Manager**. In order to collect, check and export personal data about candidates to the HUTER Platform, it was needed a secure collaboration tool. To cover these requirements, partners suggested the adoption of a case report form system to implement this feature because of being a well-known approach among researchers. “eCRF” manager is a complex tool that allow users to configure case report forms like a survey form to gather subject’s information that presumably fit HUTER Protocol statements. It also guides users during data registration process allowing the execution of validation workflows. Equally important is the data export feature that backs the HUTER Platform ingestion process.

- **Ingestion Manager**. Just like eCRF manager, it is required to check and register data uploads from partner's labs and clinics to HUTER storage. The Ingestion Manager plays a key role controlling data sampling suitability and integrity as well as tracing register for any uploaded file and request.
- **Broker – Uploader**. It was designed as a dumb tool to help labs technicians to put sample data into HUTER Storage. Broker - Uploader works coordinated with the Ingestion Manager to request submission access to the HUTER cloud. With the same intention, it also performs integrity calculations that allow the verification of files once they are loaded in HUTER cloud. Likewise, it requires to be integrated with the eCRF Manager to retrieve the case report form and upload it to the HUTER Storage. Broker – Uploader and Ingest Manager ensure that every submission process will be related to a previously registered subject, because no access will be granted if subject was not identified as suitable under the HUTER Project Protocols.

4.2. Storage

The HUTER Storage module consists of a set of tools and systems that allow partner users to save sample related data in a shared area. It also covers all requirements related to the data management: data integrity, accessibility and availability.

From the outside, this module just offers one function: file safekeeping. However, 3 complex components are needed to accomplish this goal:

- **Scalable Storage**. This component offers a flexible disk space that can grow unlimitedly according to the researchers needs. As it was said, it provides reliable mechanism for data integrity enssurance and a backup system for data availability. Accessibility is provided by the Cloud Environment from where storage is located.
- **Storage Manager**. This component means the storage core. Further ahead it will be described in detail because it is composed by many pieces. Mainly, its function is to index all data in the HUTER Platform as well as keep tracking of data by sample subject identification.
- **DICOM PACS**. Additionally, the HUTER Platform provides a DICOM Storage system also known as PACS for managing images in DICOM formats. Not only PACS performs storaging functions but also offers a DICOM compliant interface to query, upload and retrieve DICOM files.

4.3. Processing

Processing module allows users to execute analysis processes called pipelines over data stored in the HUTER platform and consolidate it again in the platform keep tracking of the original sample. Apart from analysis processes, this module will also support the transformation of images from samples in proprietary formats

to DICOM standard. It must be remembered that laboratory microscopes usually generate digital slides in proprietary formats. The current proposal was made with the intention of encourage the collaboration between clinical and research areas related to biology and health through the use of an image standard that provides image and metadata correlation.

This module also offers the functionality of creating new pipelines and defining execution environments for these pipelines. Researchers can get advantage of this feature to develop new ways of treat data through their definition under a common standard supported by HCA.

As for HUTER processing design, it mimics HCA structure base on its requirements so any tool created under the HUTER project work will be exportable to the HCA infrastructure. However, the HUTER platform must cover a smaller set of functionalities than HCA because it is focused on one project and supports few collaborators needs. Therefore, processing module has these components:

- **Repository**. It consists on 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 not only building or improving pipelines but also execute any pipeline over their study case.
- **Pipeline Manager**. This component is an orchestrator that links all needed resources to execute a pipeline on a scalable backend.
- **Pipeline Web Manager**. Since pipeline managing is a complex process it is covered by the Pipeline Manager. However, in order to ease the interaction with users, Pipeline Web Manager offers a user-friendly interface. That allows researchers to collect all needed resource references for invoking pipeline Manager. Additionally and equally important “Pipeline Web Manager” provides a secure layer that prevents users from mixing unrelated data during processing request. Furthermore, it makes possible to track every request and analyzed data.
- **Scalable Processing Backend**. For the purpose of executing pipelines, a flexible and growing on demand virtual environment is needed. Processing backend consist of an environment to execute the predefined execution environments taking as inputs the data sent by Pipeline Manager. As it was mentioned, scalable means that it is possible to execute as many parallel pipelines as needed.

4.4. 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 and retrieving.

Depending on the data type, specific tools like Image Viewer are designed. Moreover, a helping tool for exporting data to HCA Platform was added to this module.

- **Data Access Tools**. They consist on tools that allow users to look for stored data applying filters on some conditions or indexed fields. Direct data retrieving tools are included in this component to complement all the functionalities.
- **Data Export**. This component is supposed to be needed in order to execute data format transformation to fit HCA schemas and formats for data submission at the end of HUTER project.
- **DICOM Viewer**. With the intention to provide a complete DICOM solution for image management, a DICOM client is also needed. Although DICOM defines public interfaces and any viewer that supports them should be able to retrieve images from HUTER PACS, this brand-new viewer has to deal with new DICOM structures that will be proposed to DICOM Standardization Group. Must be remembered that DICOM is a medical standard so it does not currently support every image format in a research environment. So new DICOM will be proposed to deal with research environment and improve the communication between both areas.

4.5. Security, Management and Support Layer

This module involves several features that backs any IT system and it covers transverse requirements related to Data Security and Access Control.

Next components are self-described so no further introduction should be needed.

- **Identity and Access Management**. This component ensures that any access to the platform is valid. It is based on user accounts which are joined to roles. Role based security is a reliable approach widely spread in IT systems that allows managers to grant or denied access to applications and functionalities inside them. Furthermore, in the HUTER Project context, a Single Sign On module is proposed to facilitate user interaction with the applications of the Platform.
- **Registration and Discovery Server**. This component is required to provide scalability to the service layers designed. Its objective is to register and publish information about any service instance deployed in the platform. Such functionality allows HUTER administrators to scale a specific service without reconfigure other ones which consume the first one.
- **Configuration Server**. In the same way, configuration server allows administrators to on live change configurations of services using a centralized service.
- **Gateway**. On any distributed system, a gateway is needed to provide a unique entrypoint which facilitates infrastructure configuration. This piece exposes a unique entrypoint for each service and redirects client requests to a specific instance of destiny service thanks to the Registration and Discovery Server.

- **Log Management**. In order to centralize the access to the log files of every applications 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.

4.6. Dissemination & Communication

This administrative and communication module is composed of tools that facilitate team managers spread HUTER information among their teams and internet users.

- **HUTER Shared Workspace**. Bahia Software will provide an internal networking tool to allow team member to share management information as well as be used like a virtual office. So this platform should guarantee data storing in different levels of security such as among-teams information, intra-team information and per-user-role information. It also should provide tools for team collaborators organization and communication like calendars, email integration, document processor, contact list, etc.
- **HUTER Website**. As an essential tool for data publishing, a website is required to achieve a wide information spread. Bahia Software will build a website that allows any team to show their job under the HUTER project. This website will be composed of, at least, a project summary page, a participant stage section, a publication area that allows user to register their publications and a news section to add any event related to the HUTER project. It also will be used like an internal portal to unify the access to any HUTER tool just for allowed users. Regarding website look and feel, it will follow current design lines and will be accessible for any internet user.
- **HUTER Social Media**. In order to complement data dissemination, accounts on high visibility Social Media Platform will be created and integrated into the website. These tools will leverage HUTER communication needs.

5. SUBSYSTEMS DESIGN DESCRIPTION

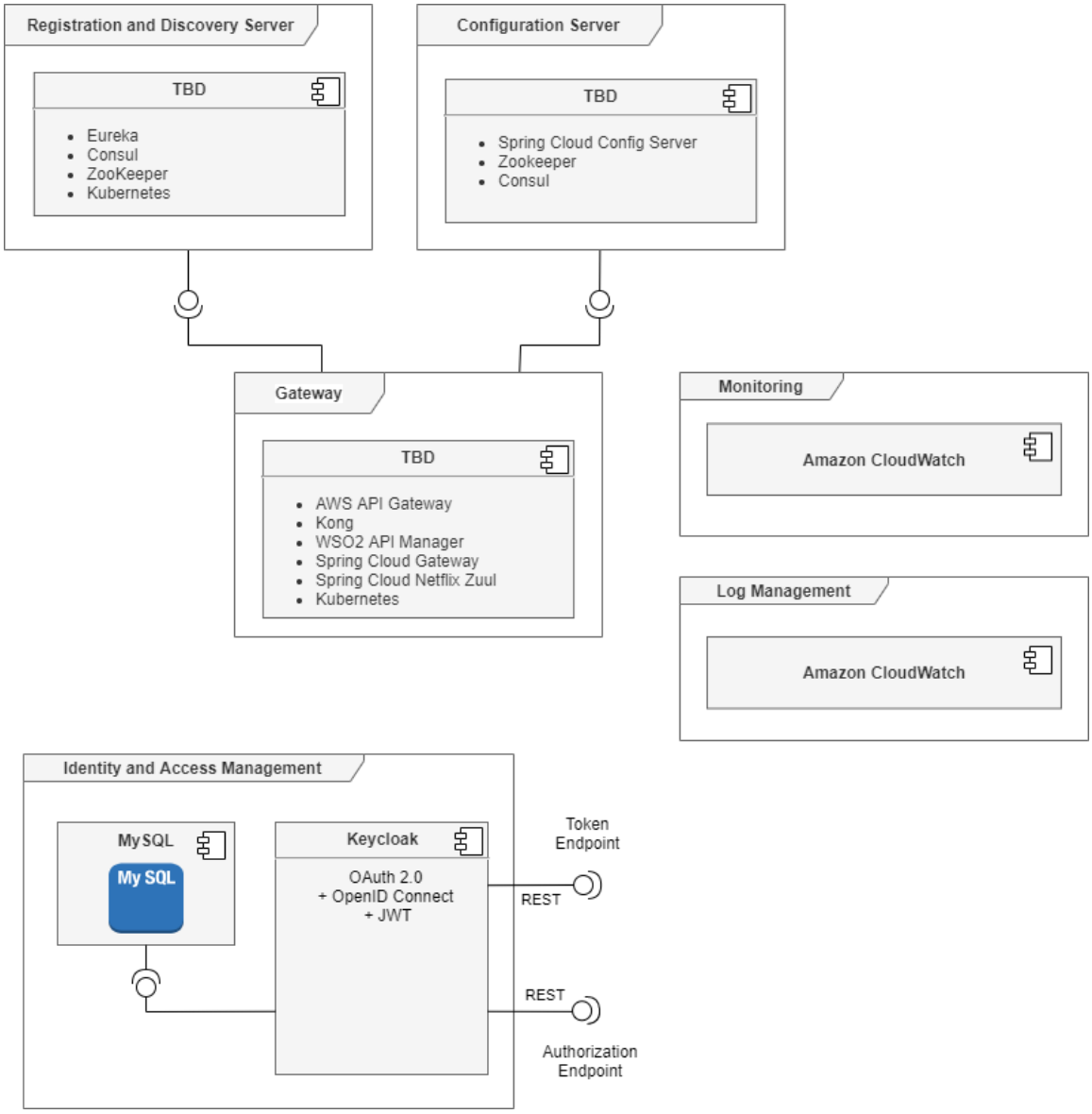
In this section a more detailed explanation will be shown focus on every subcomponent in the HUTER Platform. Current design relies on already available tools combined within brand new systems explicitly designed for HUTER. This strategy allows teams to start the data integration on the platform as soon as possible.

Equally important was the fact of lean on open source tools and technologies that guarantee the support of a large community of developers. In the same way, open source allows Bahia Software developers to face any adjustment and fix faster than depending on third-party software. This is why Open Source software has been preferred than proprietary one to cover collaborators needs on deeply studied areas such as electronic case report forms.

5.1. Security, Management and Support Layer

This module contains applications and tools that cover some of the project requirements which affect all the components. It is mainly focused on providing homogeneous methods to reach transverse functionality like user authentication or log management.

As shown in Picture 3 below, some tools are not specified because at the moment of writing this document we are evaluating different implementations to get a well-defined function. In next paragraphs, each component and subcomponents will be explained as well as the options if there is no one chosen.



Picture 3 - Security, Management and Support Layer

5.1.1. Identity and Access Management

To apply a security layer for all components in the HUTER Platform it has been decided to use an SSO (Single Sign On) approach. For this purpose, a central server will concentrate the registration of user accounts, client application and, in some cases, permissions and roles under the application.

In order to achieve this aim, Keycloak server has been chosen as a suitable solution because supplies authentication and authorization functionalities easily integrable in applications. Keycloak provides an OAUth 2.0 implementation for Authorization purpose meanwhile an OpenID Connect implementation offers Authentication functions. To get an easy integration of any application in the platform, JWT standard is implements because is based on JSON format to transfer user grants in a secure way. JWT is also able to carry user roles and additional information in its payload. Other advantage of Keycloak is that expose REST

services to integrate its functionalities in the applications. Therefore, SSO implementation in client applications becomes easy and powerful thanks to these endpoints and JWT.

Regarding to Keycloak support subcomponents, core and web application runs into a Wildfly 18 over Java 8 and uses a MySQL database in order to store user and client information.

5.1.2. Log Management

To keep track of every misleading behaviour in the HUTER Platform, an AWS Tool called CloudWatch will be deployed that enable the centralization of log registries. Cloudwath can ingest any log entry from application log systems and store it in a central repository. This way administrators can review and identify errors from anywhere in the platform just from one point. Cloudwatch also provides searching functionalities to filter log entries by several criteria.

Because Cloudwatch is an AWS service, no further infrastructure is needed to its deployment.

5.1.3. Monitoring

For platform health monitoring functionalities, AWS tools such as CloudWacth and AWS Batch Jobs Console will be deployed. These AWS tools will be described further ahead.

These tools allow administrators to measure the performance of the platform and configure alerts to advice from system yield falls.

5.1.4. Gateway

To public a stable and reachable access to the HUTER Platform is needed to expose an element called Gateway. It means a unique entrypoint to the services and tools deployed in the internal network. Furthermore, it also can route and balance petition among instances of the same requested service by users.

In current design, 2 gateways are defined: a main gateway that allow requests to access to internal network and route the requests to the proper applications; and a service gateway just for microservices access.

The first one, the main gateway, depends on the deployment platform as it will be exposed in next chapters of this document when AWS API Gateway was explained. The second one, it is not defined yet because some deployment options are still being evaluated. One of these options combined various required aspects of the microservices deployment and the other, distribute functionalities among various pieces.

5.1.5. Scalable services: Gateway, Registration & Discovery Server and Configuration Server

In order to achieve a scalable services layer, a microservice layer approach is proposed. This approach allows to create new services instances on demand to response every request. That is why this section previously shown the components explained together: Gateway, Registration & Discovery Server and Configuration Server.

As it was said in the section above, some options to achieve a scalable microservices layer are still being evaluated. The first approach is the use of Kubernetes to automatically deploy and scale services. This all-in-one solution provides not only a registration and discovery server but also a self gateway and a transparent configuration server.

On the other hand, a self made solution could be the self implementation of a microservices environment. In that case, each component (Gateway, Registration & Discovery Server and Configuration Server) must be deployed and configured explicitly to work together. Furthermore, the microservice scalability should be provided by another tools, for instance Docker Swarm or similar.

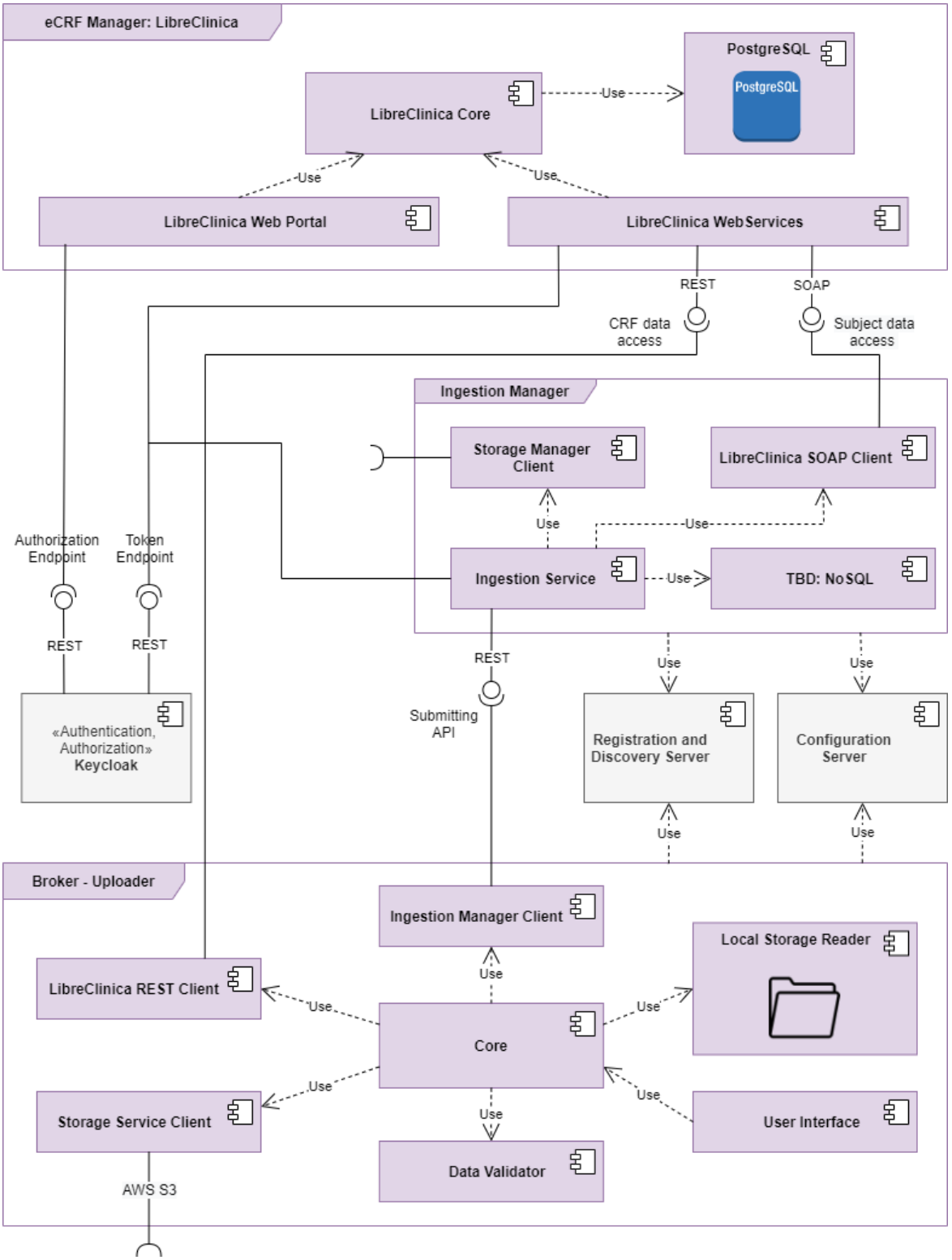
According to the existing tools that provide requested functionalities, options are listed below:

- Gateway: Kong, WSO2 API Manager, Spring Cloud Gateway, Spring Cloud Nextflix Zuul.
- Registration & Discovery Server: Eureka, Consul, Apache Zookeeper.
- Configuration Server: Spring Cloud Config server, Apache Zookeeper, Consul.

However, the purpose of this submodule is clear so the chosen option will combine the 3 named functionalities independently of the selected tool combination.

5.2. Data Ingest

In Picture 4 a detailed design of the data ingestion module is viewed. In this module several applications have been integrated to provide the data registration and data uploading to the HUTER Platform.



Picture 4 - Data ingestion design

5.2.1. eCRF Manager

As it was introduced, some functionalities have been covered by existent application and open source ones have been preferred. So as to get a complete CRF registration solution several existing applications have been evaluated but, eventually, LibreClinica v.01.00 has been selected as the more suitable application.

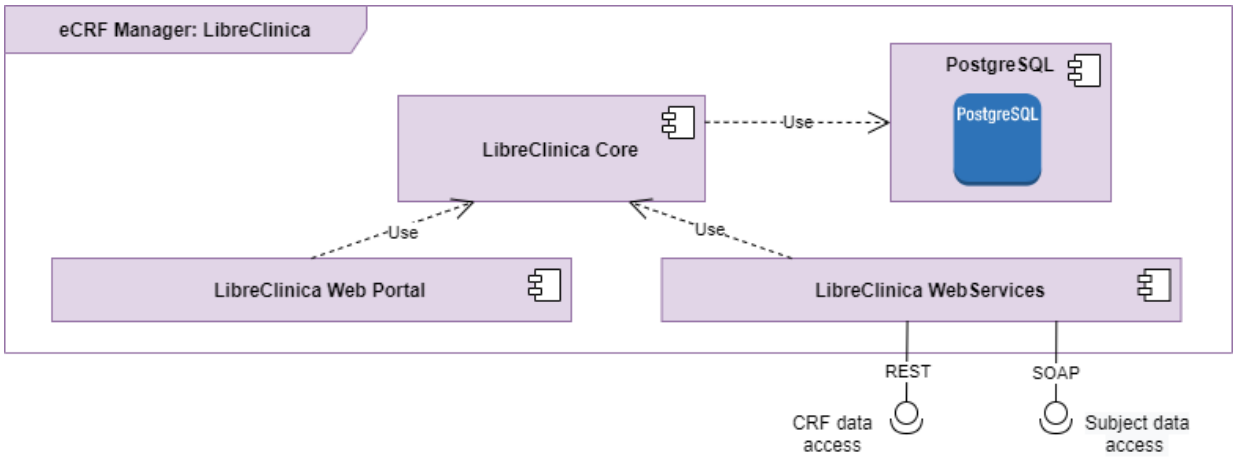
LibreClinica is an OpenClinica fork under the GNU Lesser General Public License v3.0 so free use and modifications are allowed.

It provides the capability of define and record CRF as well as queries over the question in order to apply data validation and curation. According to data access, LibreClinica provides not only a web-based interface but also a catalogue of webservices to data accessing. The web interface provides a powerful and configurable tool to perform administrative work and data registration. It also allows users to view and export data in non-standard and standard formats like ODM. Furthermore, most of these features are accessible from the webservice layer, a useful mechanism to integrate other HUTER components. LibreClinica also provides a quite complex study organization leaning on user accounts and management roles.

However, some lacks have been detected and will be fixed by improving next features:

- HUTER IAM integration. Currently, LibreClinica provides authentication against LDAP but under the HUTER project, an OpenID Connect authentication is required. To get this capability, Keycloak integration will be implemented to securize web application and webservice layer.
- LibreClinica does not provide some required features by our collaborators so it will be implemented a new subject identification and group assignment workflow for HUTER.
- Finally, in order to get a more usable application, a front-end update will be performed.

Picture 5 shows LibreClinica subcomponents as well as the technological framework that supports the application.



Picture 5 - LibreClinica subcomponents

LibeClinica is a Java based application that runs on a Java 8. It was developed and evolved applying several programming styles and patterns. However, as it is a reliable and well tested application, the most important layer to describe is Spring 5 one. This allows LibreClinica to integrate Spring Security in order to apply authentication against Keycloak and authorization using a databased roles hierarchy.

LibreClinica core keeps all the application logic exposing it through any of the existing interfaces.

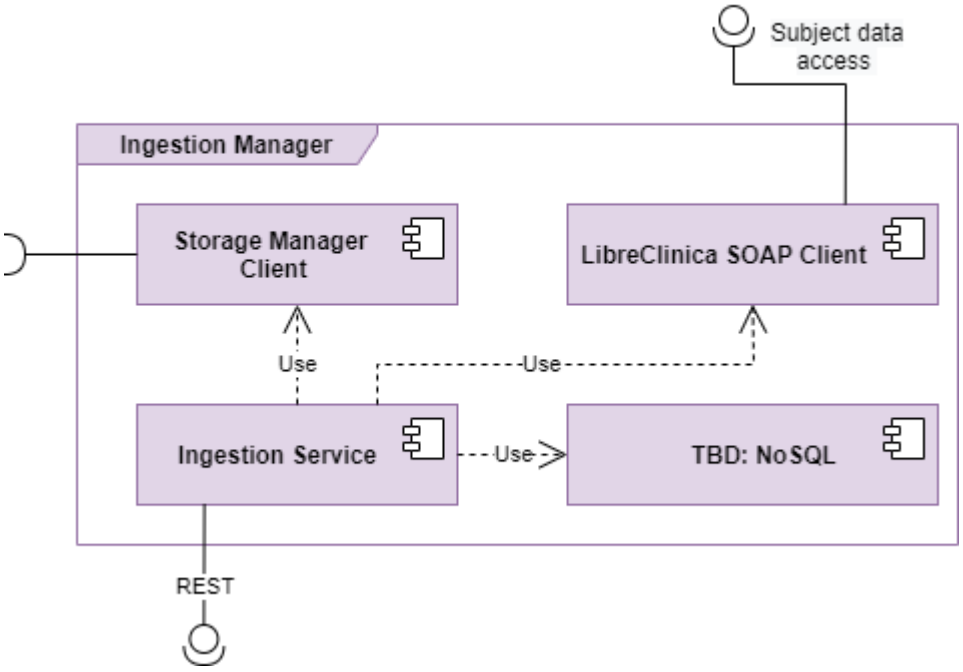
Web Portal or Web Interface was developed using JSP and Bahia Software applied a Bootstrap layer to get a friendlier interface.

Webservice layer are divided in SOAP and REST Services. SOAP ones relied on WS-Security that was removed in order to apply the Keycloak authentication and authorization. REST layer was developed under various technologies and standards like OpenRosa for shared documents.

LibreClinica relies on a PostgreSQL 9.5 database to store all the information in a relational model. The access to this database is made using a JDBC connection.

5.2.2. Ingestion Manager

This component has been designed to perform a gatekeeper role. No data submission will be allowed unless Ingestion Manager grants access after a protocol execution. To achieve the submission of a file successfully, the file has to be linked to a study subject previously registered in LibreClinica and validated by the Ingestion Manager. After that, Ingestion Manager will require data verification token in order to do a late check. If every step is completed, Ingestion Manager will provide an access token and a storage module route to upload files. After the process ends, no data will be consolidated without execute a validation step using verification tokens.



Picture 6 - Ingestion Manager subcomponents

As it is seen in Picture 6, Ingestion Manager provides a REST interface which exposes scalable microservices to manage data submissions to the HUTER platform. It is also integrated to LibreClinica user a SOAP client to check subject information and to Storage Manager through a Rest Client so as to require uploading area for submissions. To keep track of every request, it relies on a NoSQL database.

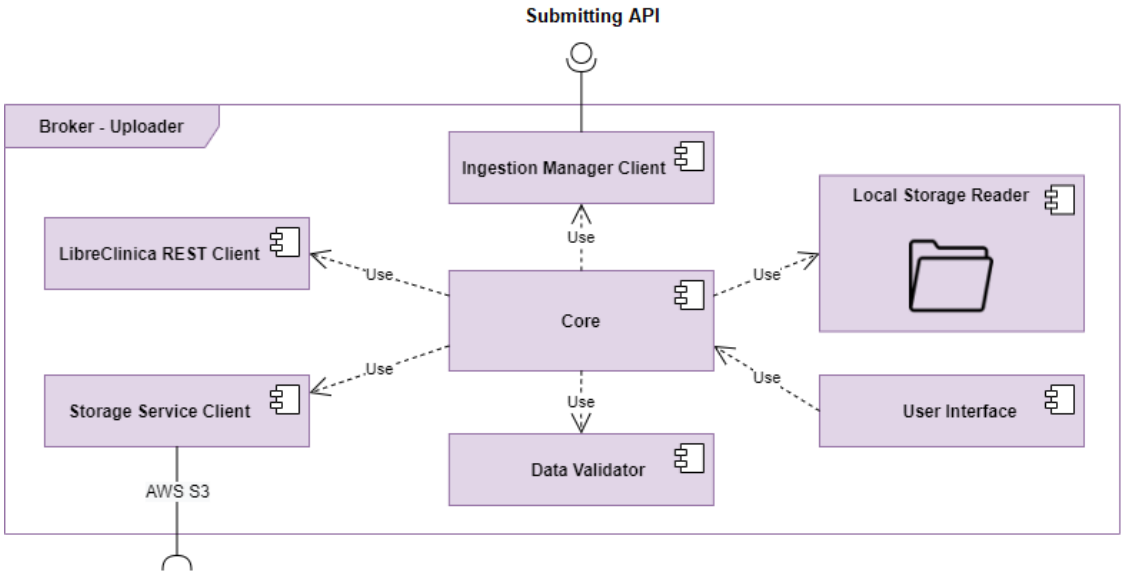
The technical framework to develop this application is Java 11 based on OpenJDK 11 because it offers new Java features, a good support of Spring Boot 2 and it is an open source environment. NoSQL database provider is no defined yet because at the moment of writing this document various options are being evaluated like MongoDB, CouchBase, Cassandra, DynamoDB, etc. The decision will depend on the yield and integration quality with the final environment.

5.2.3. Broker – Uploader

The broker - uploader shown in Picture 7 will be delivered as a heavy client and it will perform the upload of data/image files from collaborators labs infrastructure to the HUTER Platform. As it was said, Uploader will need to be integrated to Ingestion Manager in order to request submission grant for files. To get this grant Uploader should provide information about what is going to be uploaded, which kind of data is (metadata reference) and will generate some verification token through a Data Validator subcomponent. Once this task is done, uploader could connect to a specific area of Storage system and save files there. Then, after completing the task, it has to notify to the Ingestion Manager the end of the upload to trigger data validations over files in the HUTER Platform and mark it as stored.

To achieve this multi-step task, an Ingestion Manager Client, actually a REST client, will be implemented. The source of files will be the local storage of collaborators infrastructure so uploader will implement a local filesystem reader. It will also have a LibreClinica Client to obtain metadata from CRF and upload it as an additional file. Making this ingestion in such a way allows the ingestion of CRF independent from the LibreClinica WS availability. Going on listing subcomponents, a Data Validator subcomponent will generate verification tokens based on well-known algorithms such as md5sum and SHA256 which will be later used on the platform to check file integrity. Furthermore, a client to upload files will be provided and this will be tied to deployment platform, in this case, this client will implement AWS S3 libraries to upload files. Everything will be controlled in a core module which will also manage background threads in order to make uploads while users are able to keep using the uploader.

Talking about the user interface, the very first approach will be a CLI application. However and depending on each collaborator infrastructure, a heavy graphical client could be developed.



Picture 7 - Broker - Uploader subcomponents

5.3. Storage

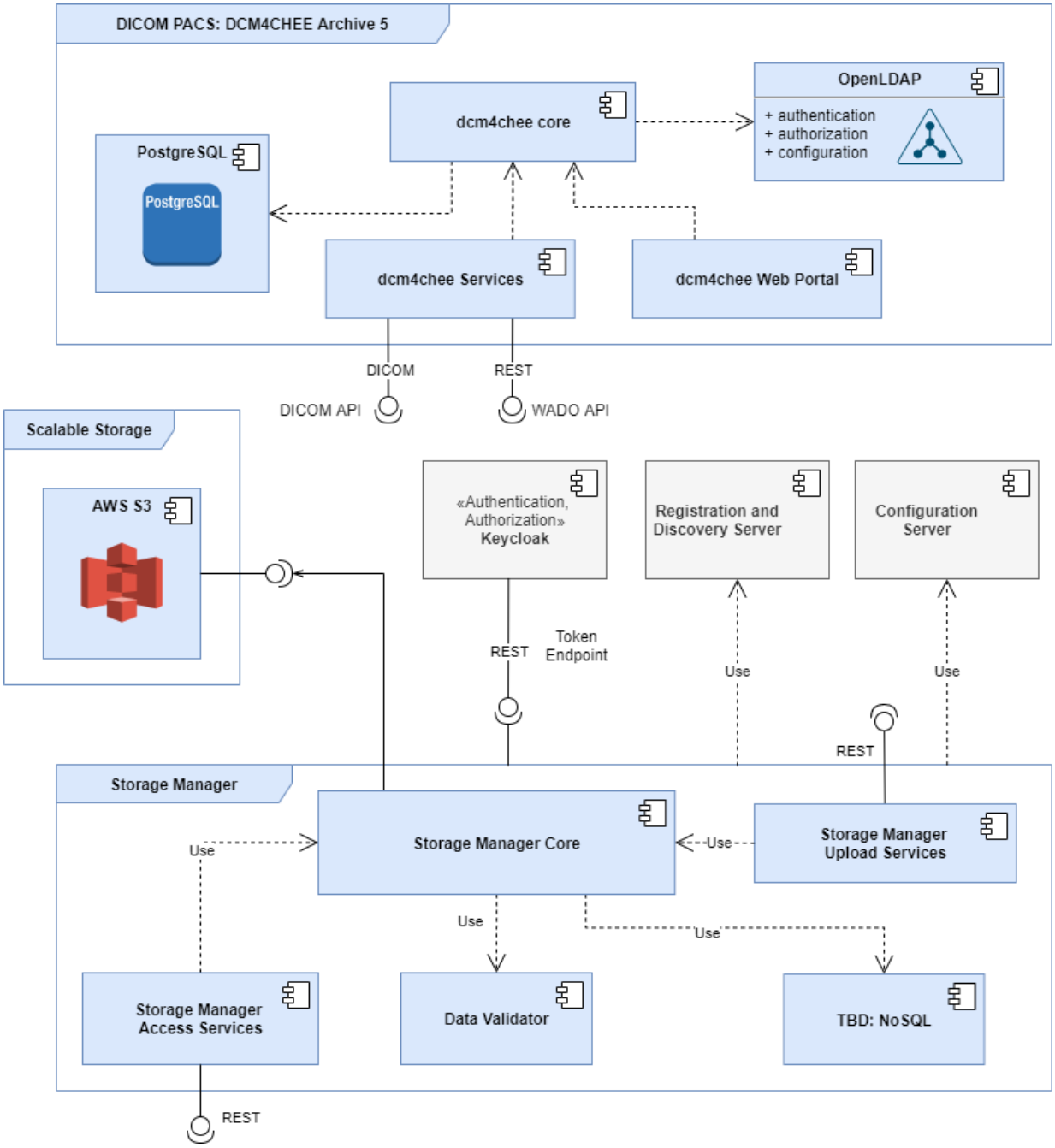
Mainly, the storage module is responsible of data storing and indexing. In order to achieve this functionality, 3 essential components detailed in Picture 8 have been designed.

To cover the previously identified requirements about storage capacity, storage component relies on AWS S3 service which provides “infinite” storage and some of the required features.

Moreover, indexing and storage managing is provided by the Storage Manager in order to keep the knowledge of what files and data are stored and the relations among them. This means that also provides information tracking functionality through these data connections.

Also, a DICOM PACS is provided as a storage complement following the Bahia Software proposal for the HUTER Project so as to encourage the collaboration between clinical and research environments.

After this introduction, a more detailed explanation of each component is shown below.



Picture 8 - Storage module

5.3.1. Scalable Storage

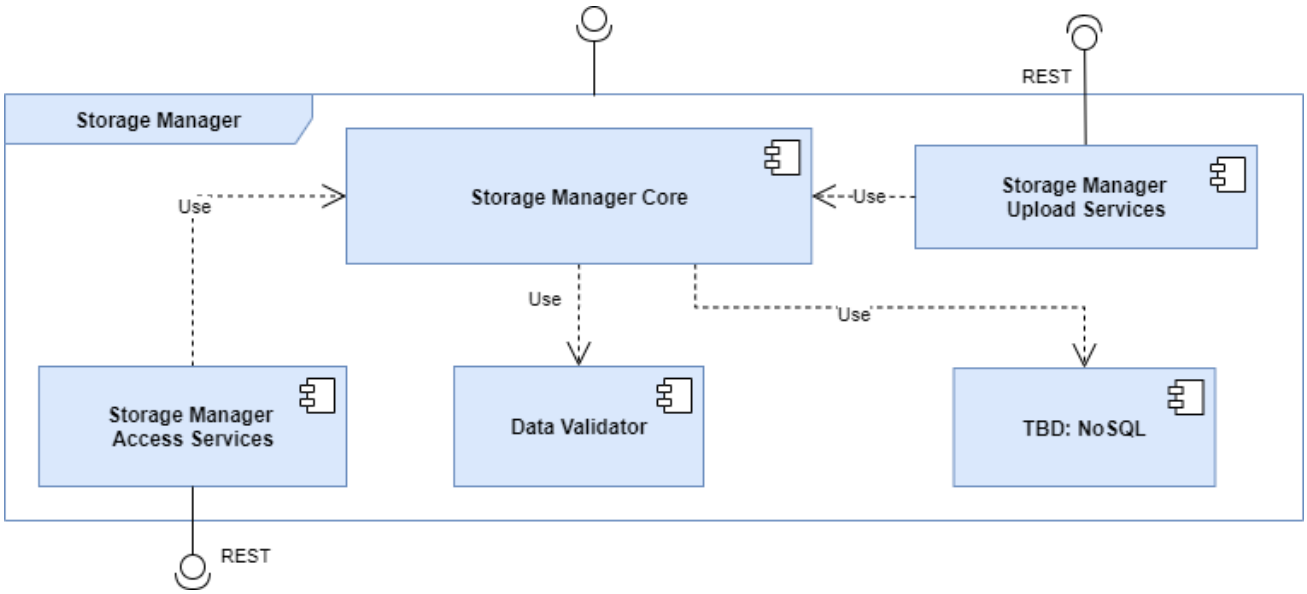
To cover the functional requirements previously identified, storage component relies on AWS S3 capabilities about scalability, availability and information security, as well as data recovery. Further ahead, in the deployment platform environment it will be explained how AWS S3 offers these features.

Regarding to stored information access, HUTER Platform clients could access it through an AWS S3 client developed using a proprietary library. This client should be integrated in any application that needs access to stored files or data like some components in the HUTER Platform. It is important to point out that using this

client a security layer is added because any access through it requires a previous grant access and, as it was previously said, this permission is given by HUTER services and IAM services.

5.3.2. Storage Manager

This component provides a management core for the stored files and data. 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, etc. (see Picture 9).



Picture 9 - Storage Manager

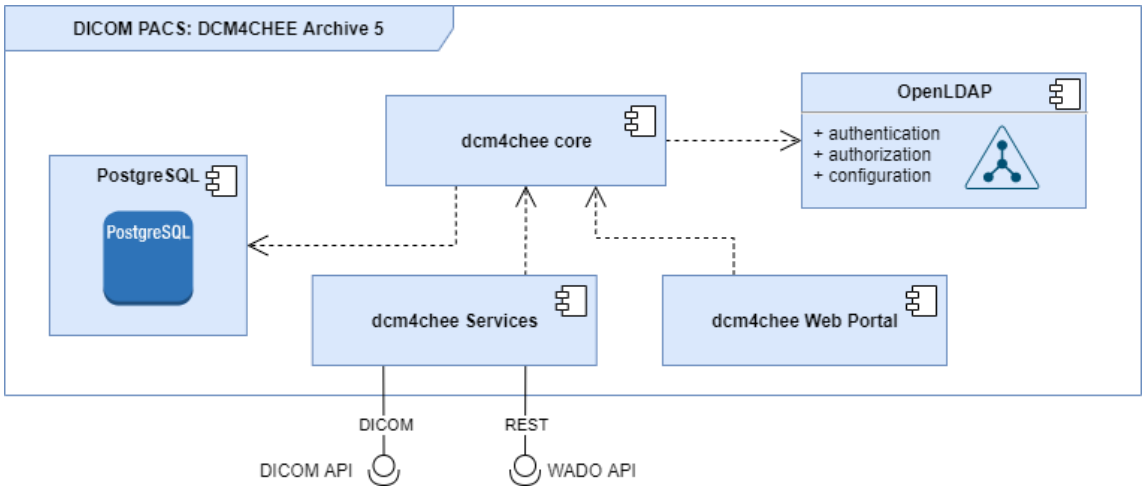
Besides, it exposes two layers of microservices: one for data consolidation and other for data access. Both layers follow the microservice philosophy over REST that gives them scalability and reliability. According to the security, they are integrated with de IAM, so only granted users could access to them.

Talking about the data consolidation services, it covers all the functions required to organize upload requests through the Ingestion Manager and the results of the Processing Module. To ensure a proper quality and integrity of the stored files, a dedicated subcomponent for file and data validation (schema, md5sum, SHA and other) is integrated and invokable through the service layer.

Finally, to give support to the indexing and querying, all data will be stored in a NoSQL database. At the moment of writing this document, no database brand has been chosen yet, but the approach is clear.

5.3.3. 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.



Picture 10 - DICOM Archive and Communication

In Picture 10, the complete architecture of a standalone server is shown. No complex and high availability system will be provided in that case because this approach has a dissemination and a first effort intention to bring closer clinical and research worlds.

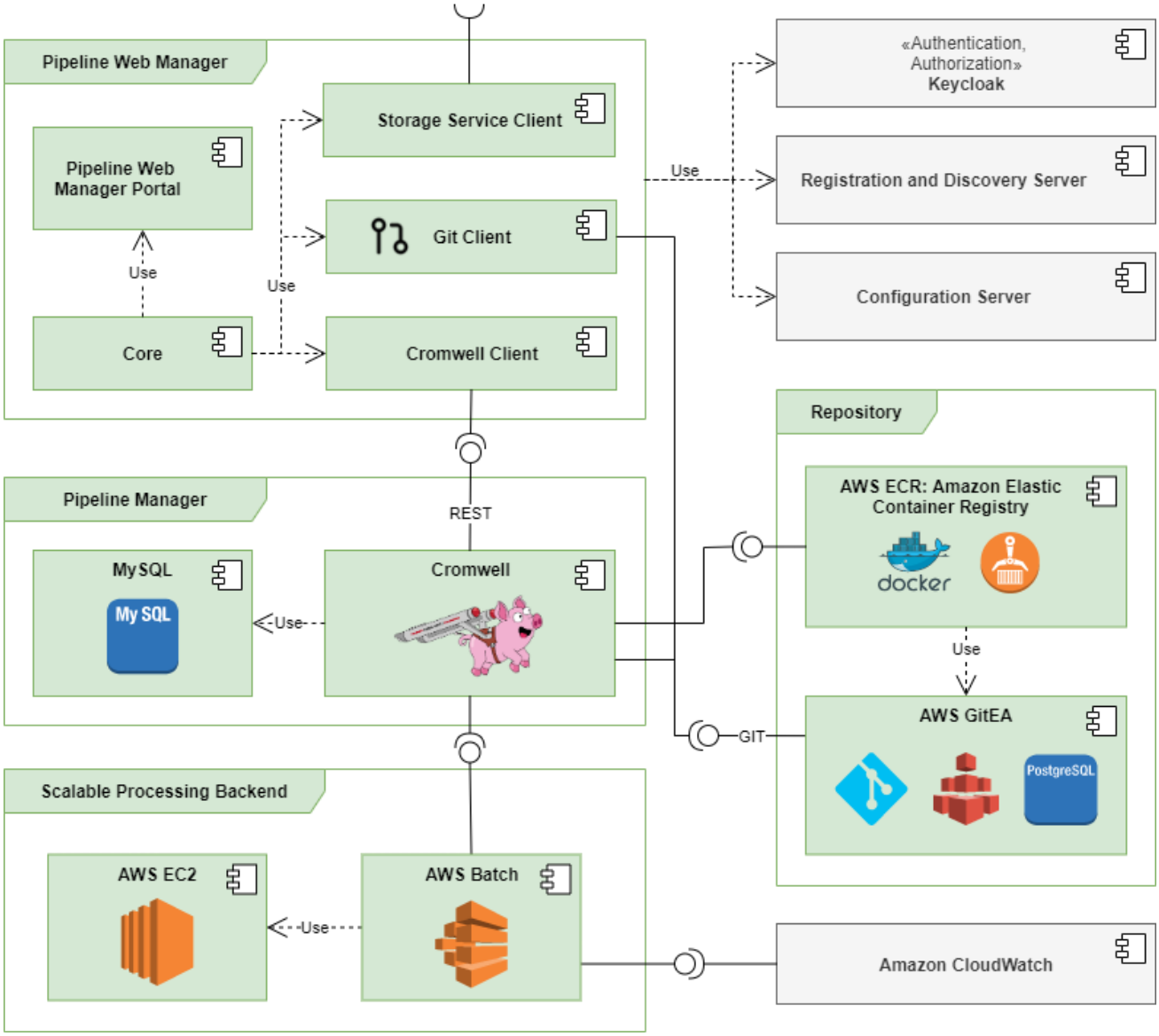
Getting back to the technical solution, Bahia Software has chosen an existing implementation of a DICOM PACS called DCM4CHEE Archive 5. This solution has an Open Source nature and its reliability has been tested in previous projects.

Subcomponents that are required to deploy for running a DCM4CHEE are:

- Dcm4chee core which concentrates the management logic for image storing and metadata access.
- OpenLDAP server to keep any dcm4chee configuration.
- PostgreSQL database for metadata and files indexing propose.
- Dcm4chee web portal for administrative tasks.
- Dcm4chee Service Layer for data storing, data access and data management tied to the DICOM. DICOM standard identify some features that PACS should provide through these services as well as the way in how they should be exposed. Both exposure protocols are based on directives over data in order to store, erase, move or retrieve them using a REST interface or DICOM command over a socket communication.

5.4. Processing

Processing module should allow collaborators to run analytical tools over the data ingested in the platform. Thus, no external files could be included unless they were ingested and validated. Once these tasks have being performed, they will be selectable as inputs in the pipeline execution process. The result of this analysis will be ingested in the platform automatically linked to the input data in order to keep the information tracking. In Picture 11, the complete design of processing module is shown.



Picture 11 - Processing

Information tracking will be reachable thanks to the integration against the storage module through their service layer.

This module also considers the possibility of generating new pipelines definitions and executions environments as well as using them in the analytical process. New pipelines could be developed using WDL

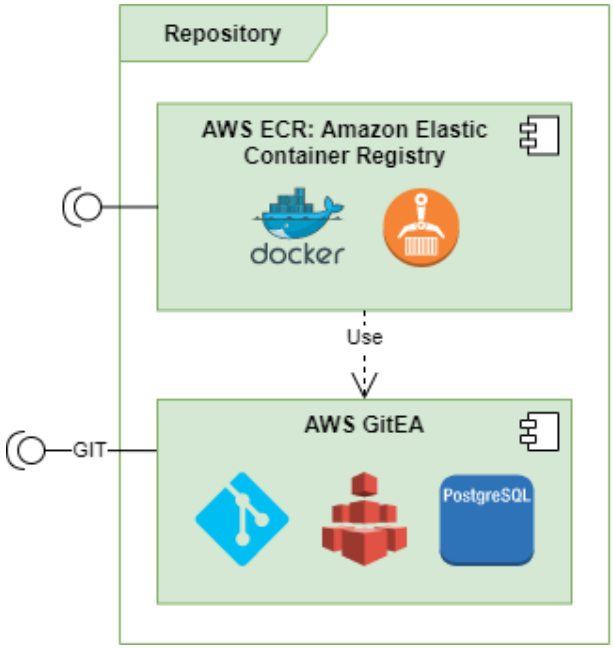
format, following the HCA guidelines, so it will be exportable to the HCA in the future. If new execution environments were needed, docker container images or their definition as dockerfiles that would be automatically transformed in a docker container image should be provided.

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 the current DICOM standard or a proposed approach if none exists.

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. In the next section, each component will be detailed.

5.4.1. Repositories

Regarding to collaborators needs, two repositories have been created to develop and create tools for data processing. Picture 12 demonstrates the definition of both of them and the relation between them.



Picture 12 - 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.

5.4.2. Pipeline manager + scalable processing backend

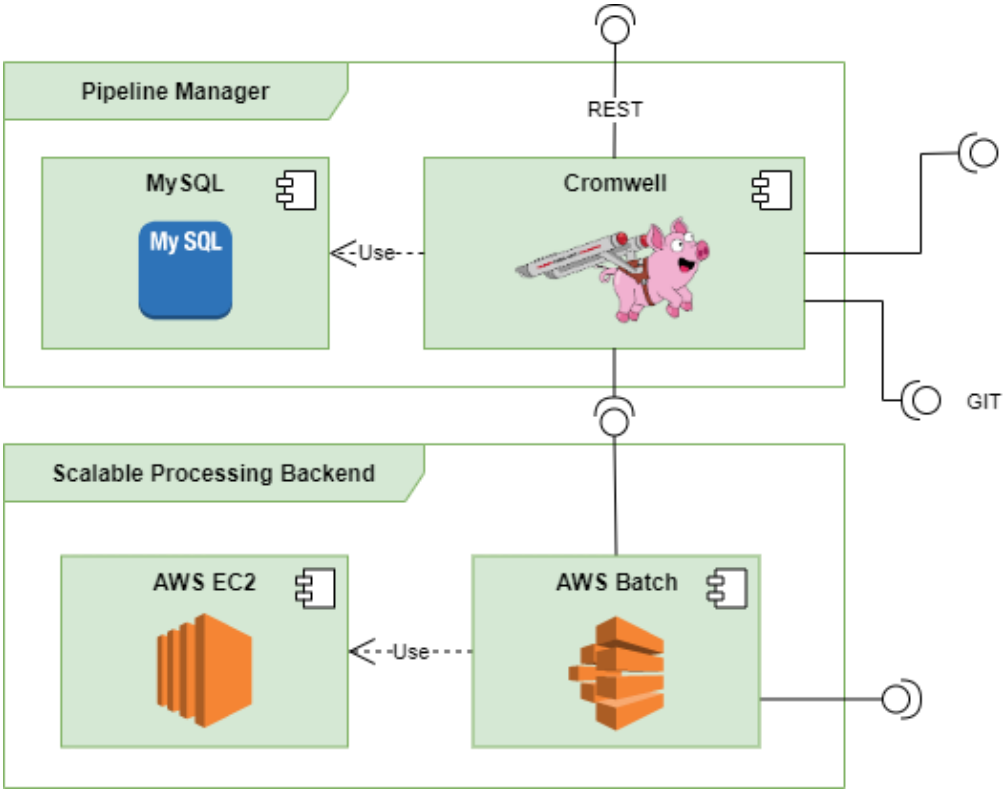
These two modules will be explained together because both are necessary to achieve data processing.

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's 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.

In Picture 13 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.



Picture 13 - Cromwell complete infrastructure

Cromwell is such a flexible environment that can also be used to integrate the dicomization process proposed by BAHIA. As it was said, this that should obey same rules of traceability as the rest of the data processed in the platform. So specific WDL and docker images can be defined to execute dicomization processes.

The most important features to point out are that:

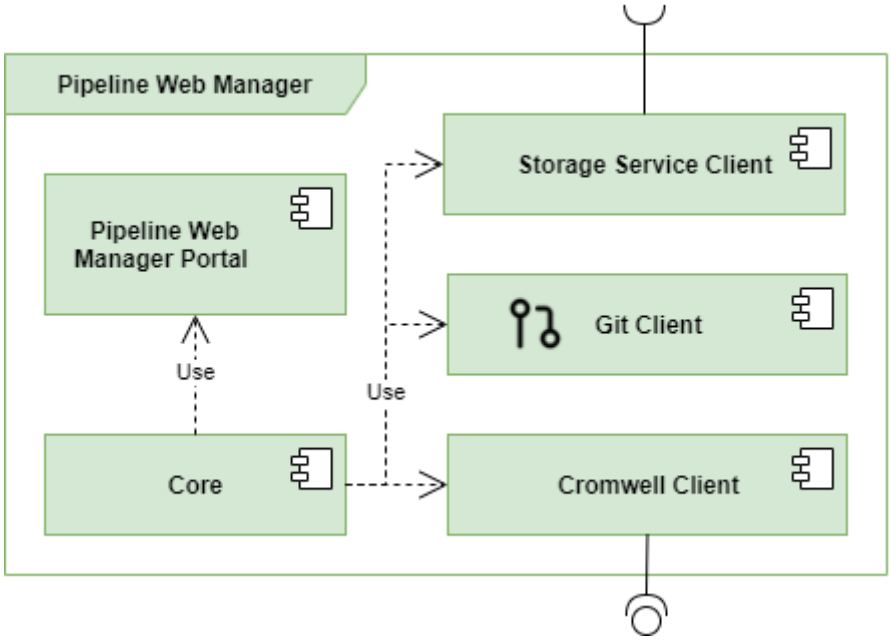
- Cromwell instance runs over a MySQL database.
- Cromwell is also integrated with AWS ECR and git to retrieve both WDL and docker image reference to invoke the backend.
- Processing backend is supported by AWS Batch that provides a scalable system of AWS EC2 machines to run docker machines.
- Cromwell 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.
-

5.4.3. 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 to facilitate Cromwell executions but to

ensure data tracking. The provided features allow HUTER Platform users to access not only to Cromwell but also to a list of existing pipelines which have 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 ensure right data correlation.

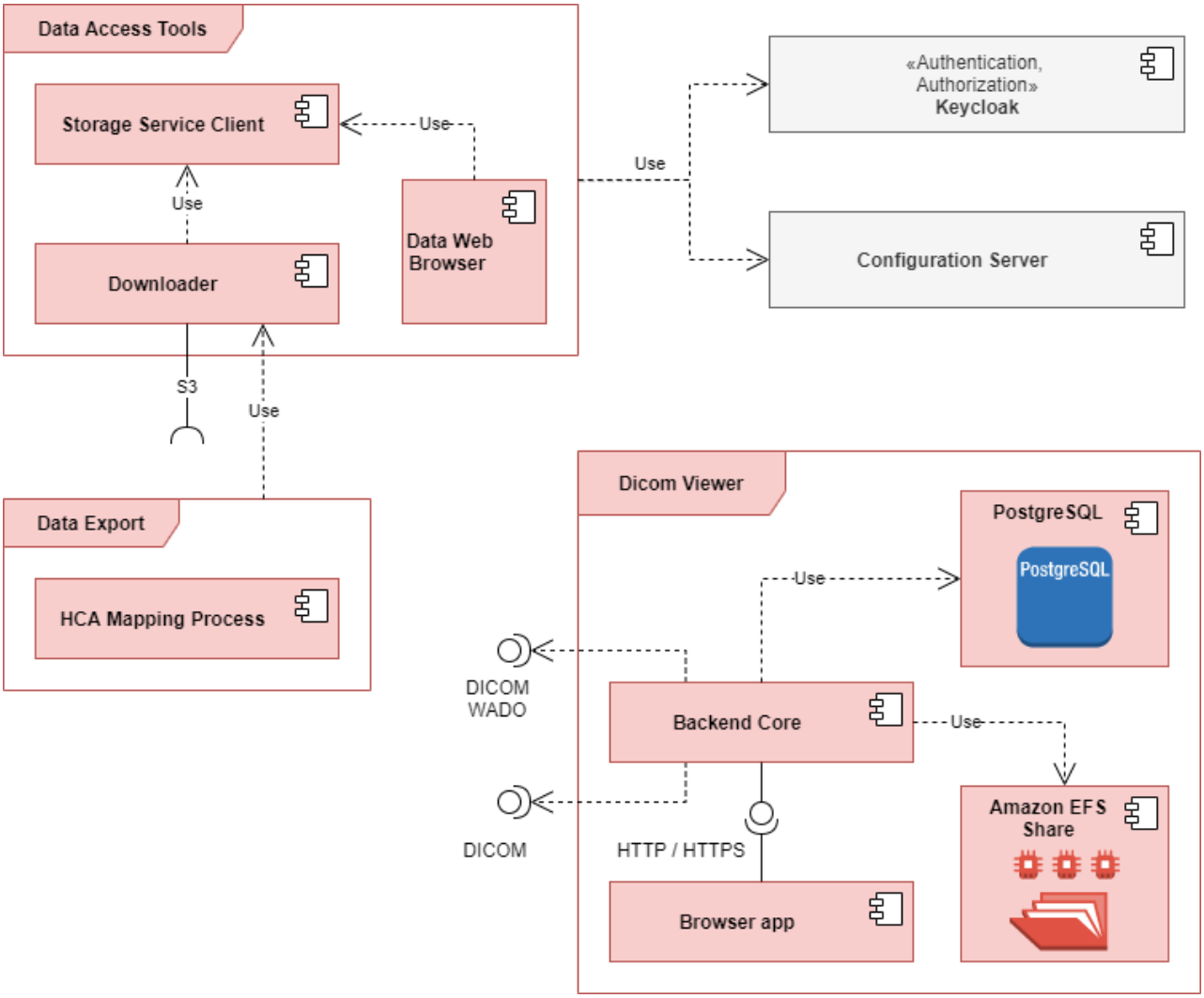
In Picture 14, Pipeline Web Manager components are shown. The main logic is contained in a core that exposes an interface through a web portal integrated to HUTER SSO. Core subcomponent performs an integration role among stored files thanks to a Storage Service REST client, a git client pointing at gitEA and a Cromwell REST client to consume its interface and invoke the pipeline execution schedule.



Picture 14 - Pipeline Web Manager

5.5. Data Access

Data Access module comprises a set of client applications and resources provided by the HUTER Project to access to HUTER Platform. Picture 15 shows the general composition of this module which is a set of different applications that could be integrated between them or run independently.



Picture 15 - Data Access module

5.5.1. Data Access Tools

In this pseudo component HUTER Platform clients are grouped because all of them are expected to be tools allowing users to query, visualize or download stored raw data and metadata. In previous sections, it has been introduced the need of securing data access through IAM server and a service layer for data querying, so these client applications are compliant to this accessing method.

Before go 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.*

A list of the subcomponents and applications is shown next.

5.5.2. Storage Service Client

This subcomponent is not an application in itself but it is an embeddable piece that facilitates communication between client applications and storage manager under the REST approach. Integrating this subcomponent in any client, it will manage the procedure to get access to the platform based on each client configuration as well as provide a close API to be invoked from client applications.

It must be remembered that the security access is provided by Keycloak using OAuth 2.0 for authorization, OpenID Connect for authentication and JWT as transport standard. In that way, next components will be able to communicate to the HUTER Platform.

It will be developed under the technology and framework described further ahead but it is expected to be released as a java library configurable from the host application.

5.5.3. Data web browser

This component is a web portal that allows user to query information in the HUTER Platform. This application 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 could be used to download raw data.

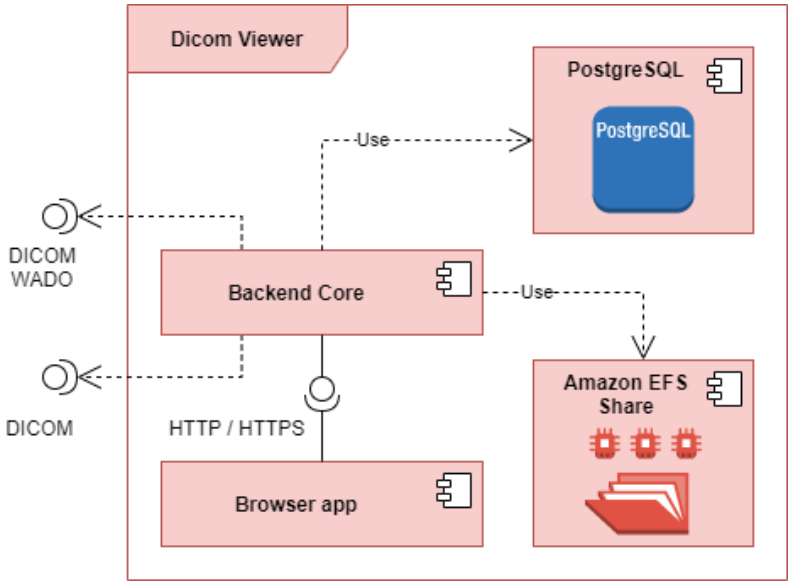
Communication against the HUTER Platform is made thanks to the integration of the Storage Service Client.

Talking about the development technology, this will be a Spring Boot 2.0 web application with no database because any data will be retrieved through services. Regarding web interface, JSP technology will be used as the basement but dynamic front end component built with Vue 2 JS are also expected.

5.5.4. DICOM Viewer

DICOM Viewer it is a necessary complement to the DICOM PACS because, it has been said before, DICOM it is not just an image format but also consider a communication standard. So as to get the capacity of view stored DICOM images this tool will be needed.

Picture 16 shows the subcomponents and communication among them and the expected DICOM interfaces that it consumes for PACS communication.



Picture 16 - DICOM Viewer subcomponents

For the building process of this tool, Bahia Software has collaborated to Sixtema because of their experience in graphical mapping. Thanks to that, they can develop a tool over the DICOM standard able to visualize microscopy images like a tiled pyramidal map.

Technological environment is slightly different from Bahia Software one but they share Java as the base development technology. Frontend app is built using AngularJS 1.5 and the core application lean on a postgreSQL 9.5 database. Additionally, a filesystem is used like a cache in order to reduce the latency communication with the PACS.

5.6. Data Export

As for Data Export, it will be a tool for non-technical 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.

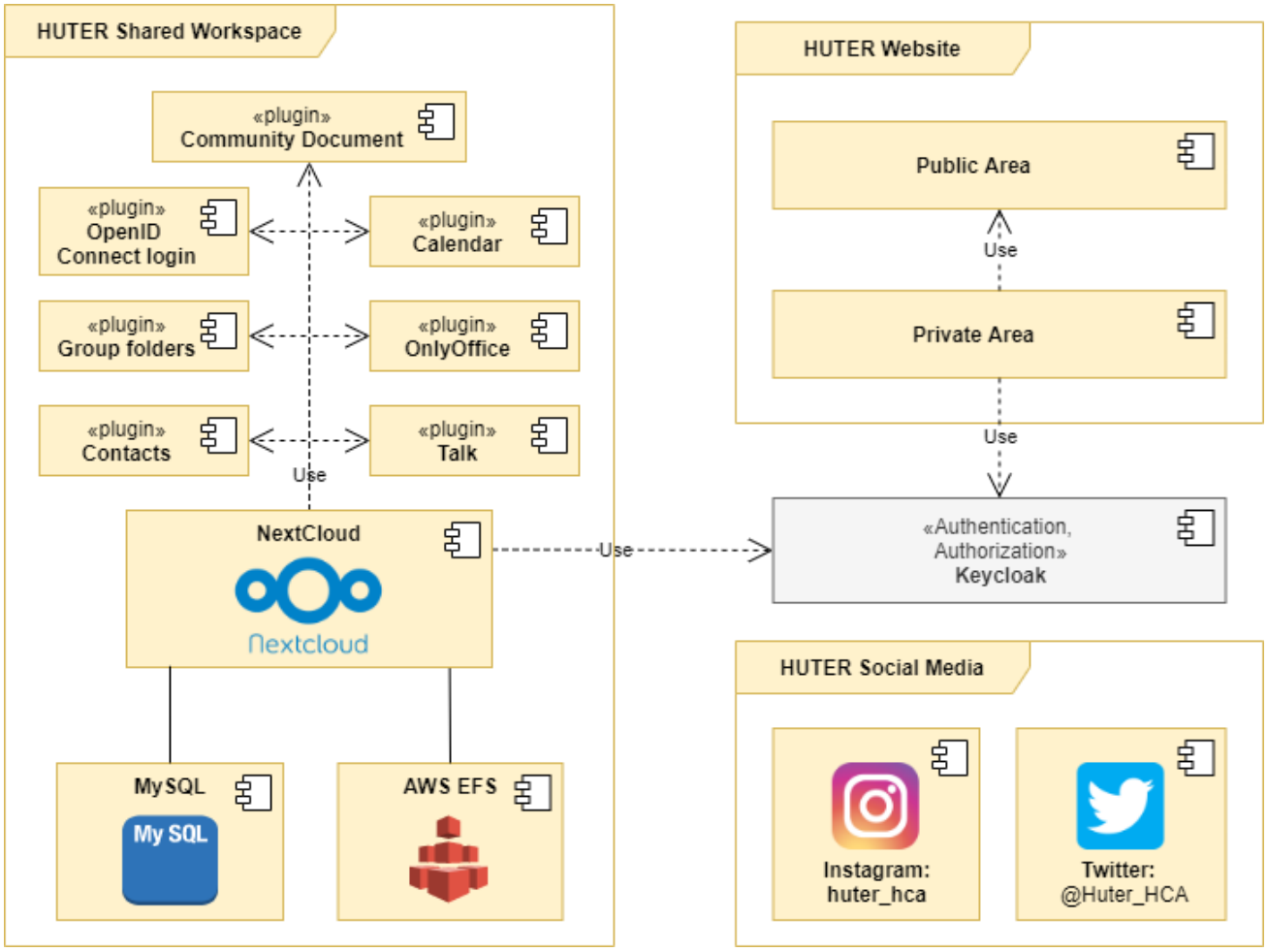
In order to facilitate data analysis, the HUTER Platform will storage data and metadata in the most suitable format for HUTER collaborators. However, it will always be taken into account that the content fits the HCA DCP guidelines.

So as to translate the metadata information into HCA JSON schema compliant structures, this additional tool is expected to be needed.

5.7. Dissemination & Communication

The dissemination and communication module is composed of 3 independent elements. Each one covers complementary needs of HUTER Project managers.

Next section explains the components shown in Picture 17.



Picture 17 - Dissemination and communications components

5.7.1. HUTER Shared Workspace

Collaborators requested a data sharing platform to delivery common data among the HUTER Project members. In order to facilitate this space and keep the integration within the platform, a NextCloud installation is provided to all the collaborators.

Nextcloud was chosen because it allows the integration of several plugins to offers a wide set of additional functionalities. In Picture 17, installed plugins on Nextcloud were listed and they will be explained next:

- OpenID Connect login: it supports the integration with the IAM server Keycloak.

- Group folders: it allows the administration and folder permission over groups of users. This feature was required in order to create folders which can be shareable among different groups of HUTER Platform users.
- Contacts: it offers a personal registry by user to keep a list of people and their contact methods: mail address, phone number, etc.
- Talk: a plugin that provides videoconference capabilities inside the HUTER Platform.
- Calendar: this plugin offers a calendar that can be synchronized with other calendar applications like Gmail using WebCAl protocol. It also allows to send event invitations to other users.
- OnlyOffice: it is a suite that works like a rich text editor so users can create and edit their document online.
- Community document: it is a plugin that deploys a document sharing server in Nextcloud allowing the collaborative modification of OnlyOffice files.

Additionally, Nextcloud integration with Keycloak through OpenID Connect login allows users to take advantage of SSO features to access platform facilitating the usability of this internal communication space.

5.7.2. HUTER Website

Knowledge dissemination is one of main targets of both HUTER and HCA project, so the creation of a website to publish HUTER Project progress is essential. With this intention, a website was designed to publish information about the HUTER Project, its participants, attended events and project collaborators achievements.

So, this website was developed using HUGO framework because it helps in building static websites isolating the data from the structure of the website. Furthermore, it covers the generation of all type of data publication needed: news, collaborators publication list and static informational pages.

It also accepts the use of themes, what are a set of html templates and resources (css & js) that simplifies interface generation and behaviour injection.

Because Javascript files delivery is possible, specific website behaviour can be developed. Taking advantage of this capability, some javascript components have been integrated to add an attractive look to the website such as carousel effects.

According to the next point, the HUTER Project will have social media accounts. Twitter feed can be easily integrated in the HOME page of the website through a lightweight javascript client.

Apart from javascript support, HUGO offers some features that help in the creation of blog-like websites. This means that facilitates the addition of content and supports the expected behaviour of a news board even applying tag and categories to them. This feature was used to generate the collaborator's publication

area and the news section that can be used by existing widgets to automatically publish them in recent-news widgets or search them in a tag/category browser.

Previous information about website was related to a public area but a private area just for HUTER Platform users has been designed in order to provide a HUTER data portal. From this area which will be secured by a Keycloak javascript adapter and generated using Vue 2 JS components, users will be able to access to HUTER Platform tools. Thanks to the SSO feature, users will just need to login once from here to access anywhere in the platform if they have been granted access.

5.7.3. HUTER Social Media

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

6. COMPONENTS DISTRIBUTION

In this section the deployment architecture of the HUTER Platform will be shown according to the application design which covers research's needs. Some components of this architecture have already been introduced in previous sections when software solution was explained.

To get started, a high-level view of the infrastructure will be explained and then, a more detailed explanation of each component deployment will be shown so as to check single details. Not all components have a defined deployment model yet, like mentioned services. It must be remembered it is being evaluated some deployment models for these services.

Before going on introducing the deployment model, it must be highlighted that Amazon Web Services, AWS, has been chosen as the infrastructure provider. AWS is one of the few platforms that can deliver an infrastructure able to give the request resources by the HUTER Project like "infinite" storage space and on-demand-scalable processing yield. This was the main reason for selecting AWS to deploy HUTER Platform. Additionally, other features of AWS were taken into account in order to be chosen:

- On-demand resources scaling.
- Robust solutions which are adapted to different needs.
- Easy integration of solutions.
- Fast configuration of new resources and deployments.
- Secure environments.
- Availability, Consistency & Reliability ensurance.
- Storage data protection, Automated Backups and Streamlined Disaster Recovery.
- Ready-to-use management layer.

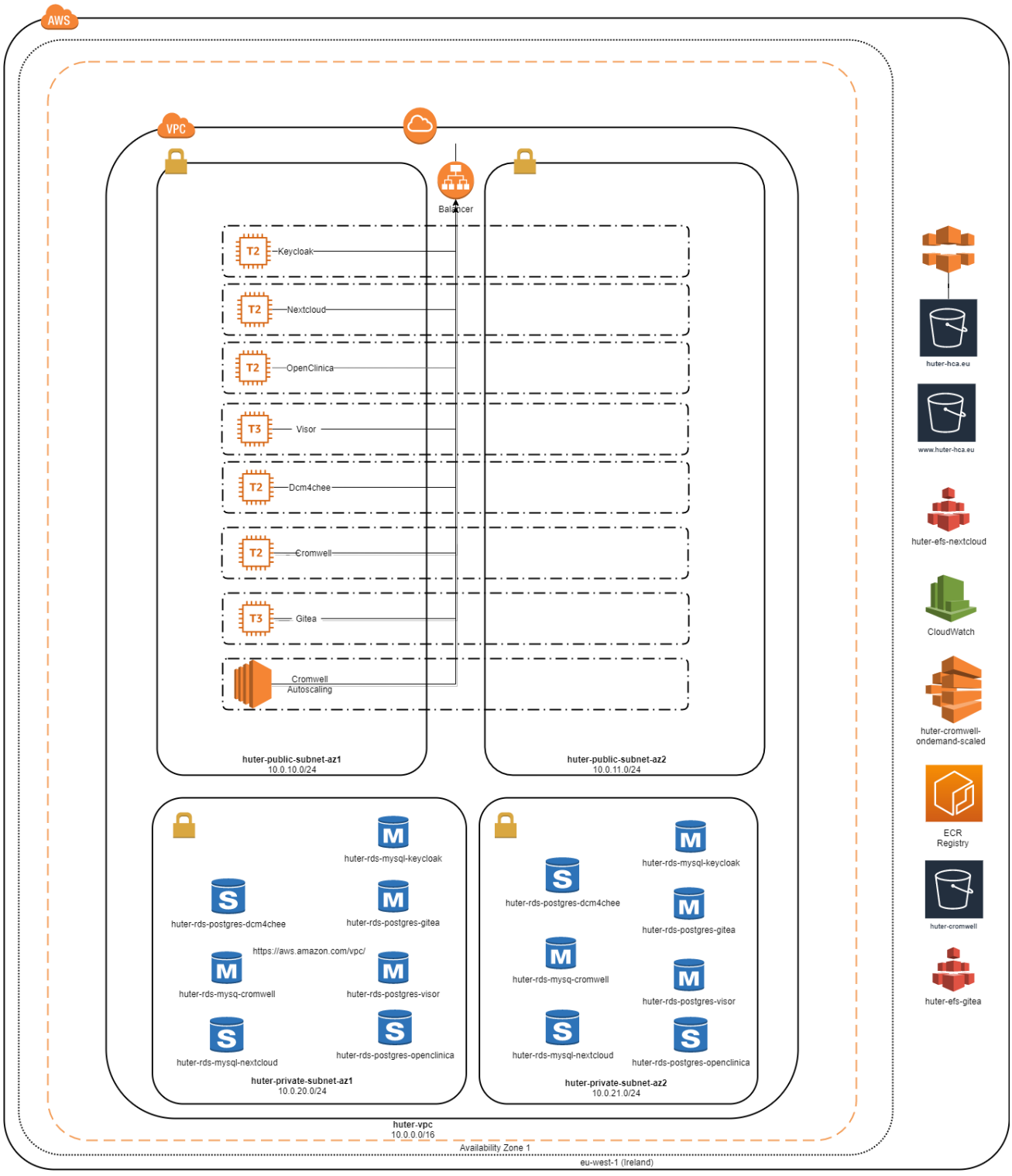
Not only technical features were measured but some legal considerations were considered. In the Ethical Plan the restriction of subject information sharing outside the European borders had to be obeyed so no deployment platform could be out of this limit. Therefore, AWS premises in Ireland was preferred to build the HUTER Platform.

6.1. Deployment model

In Picture 18, the current deployed infrastructure is shown. According to this schema, new machines could be created in a similar way so as to serve any of the required application previously introduced in the SUBSYSTEMS DESIGN DESCRIPTION section.

In order to focus each section of this document on a particular aspect of the HUTER platform, the first introduced aspect will be the network configuration, followed by the application server where applications are deployed, continuing to the databases infrastructure, the stage system and finally, the management

tools. In each section detailed information of the current used services and tools will be explained and furthermore, how it will grow until reach the final architecture.



Picture 18 - Current deployment architecture

6.1.1. 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.

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.

There have also been configurated two subnets for the RDS Service that supports database manager. Further ahead RDS will be detailed so for now the network schema it is the only space considered in this section.

Regarding 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.

Regarding general considerations of networking, the expected protocols to be managed are all over TCP/IP. According to the application protocol most of the communications will work under HTTPS but some applications will use other protocols to access to specific services:

- DICOM protocol to store or retrieve DICOM objects.
- S3 over HTTPS for raw data accessing.
- SSH for management purposes.

6.1.2. Application servers

Most of the current machines configured to support the deployed applications are instances of T2 Amazon EC2. T2 instances are low cost machines with limited yield that can be grown until the 32 GB of RAM and 8 virtual CPU.Once the completed HUTER Platform is up and running, the top characteristics will be upgraded.

There are other applications that need a high performance because of their purpose which are already running over T3 Amazon EC2 instances. These machines have a non-limited initial yield so they are suitable for important services or those which need to have a stable performance.

If more applications were needed, and they are planned to be, more machines T2 or T3 would be configured.

Additionally, in some cases, autoscaling AWS service will be needed in order to provide an acceptable yield depending on the service demands. For that purposes 2 different services are expected to be used:

- AWS Batch: previously introduced as the backend manager for Cromwell executions. It will allow the instantiation of EC2 machines to run pipelines into docker images.

- Amazon EKS: Amazon Elastic Container Service for Kubernetes, which is a service that handles microservices requirements on-demand. It can allocate EC2 machines for new microservices instances deployment and the automatic registration and exposition though a gateway. Even though this service is not configured yet, it is expected to be the supporter for the microservices layer of those components that expose them.

6.1.3. Databases

As it is shown in Picture 18, several instances of different relational and NoSQL database engines are required. Each kind will be deployed on a different AWS Service but at the moment just some of the designed relational databases are running in the HUTER Platform.

6.1.3.1. 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 - optimized for memory, performance or I/O - and provides six familiar database engines but HUTER Platform will just need PostgreSQL 9.5 and MySQL 5.

6.1.3.2. NoSQL

In order to deploy a non relational Database, AWS offers several options depending on the required engines. At the moment of writing this document, no database type has been chosen yet for this purpose so in the final architecture document this section will be completed.

6.1.4. Storage

HUTER Platform will need different types of storage systems depending on the characteristics of the data to be stored. The list of used services for storage is AWS S3, AWS EFS and local machine storagement. Next sections will explain each one and their features.

6.1.4.1. Local machine filesystem

In previous diagrams, application servers have been explained but nothing was detailed about the local storage needs of each one. That is why storage features can be modified on demand depending on their needs. However, no large local storagements are expected, just small local spaces for working spaces with configuration or small temp files.

6.1.4.2. AWS EFS

Amazon Elastic File System (Amazon EFS) provides a simple, scalable, fully managed elastic NFS file system to use with AWS Cloud services deployed in the HUTER Platform. It is built to scale on demand to petabytes

without disrupting applications, growing and shrinking automatically as you add and remove files, eliminating the need to provision and manage capacity to accommodate growth.

AWS EFS service has been configured in case of sharing needs among various application instances but will not be used for standard storagement.

6.1.4.3. 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 extremely durable, highly available, and infinitely scalable data storage infrastructure.

AWS S3 service is used to store heaviest files in the HUTER Platform. Its organization is controlled by the storage manager directly or through the Ingestion manager using AWS Buckets for separating modules spaces. However, the direct access will be done using the AWS SDK or the AWS API REST in multipart mode layers ensuring data protection.

Data protection refers to protecting data while in-transit (as it travels to and from Amazon S3) and at rest (while it is stored on disks in Amazon S3 data centers). It is able to protect data in transit using Secure Socket Layer/Transport Layer Security (SSL/TLS) or client-side encryption. The following options for protecting data at in Amazon S3 are considered for uploading to the HUTER Platform:

- Server-Side Encryption – Request Amazon S3 to encrypt object before saving it on disks and then decrypt it when they are downloaded.
- Client-Side Encryption – Encrypt data client-side and upload the encrypted data to Amazon S3. In this case, client applications manage the encryption process and the encryption keys.

Storage will always be done using server-side encryption. Amazon S3 encrypts data at the object level as it writes it to disks and decrypts it when it is accessed. Main advantage of this method, apart from data security, is that there is no difference in the way you access encrypted or unencrypted objects. Despite encryption, listing objects from AWS buckets will return all objects regardless of whether they are encrypted. Therefore, this feature ensures data security without obstructing client behaviour.

AWS S3 provides 3 methods to manage certificates used during data encryption and for the HUTER platform has been applied the Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3).

6.1.5. Out-of-the-box AWS Services

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

6.1.5.1. 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 4.3 Processing as the backend for Cromwell executions that needs to run WDL over docker images.

6.1.5.2. 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.

6.1.5.3. 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 integrated with the Gitea solution to automatically build docker images based on the pushed docker-compise files, simplifying the creation of new pipeline execution environments.

Amazon ECR eliminates the need to manually operate a container repository or worry about scaling the underlying infrastructure. Amazon ECR hosts docker images in a highly available and scalable architecture, allowing HUTER researchers to reliably deploy containers for pipeline execution. Integration with AWS Identity and Access Management (IAM) provides resource-level control of each repository.

6.1.6. 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.

6.1.6.1. 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.

CloudWatch collects monitoring and operational data in the form of logs, metrics, and events, offering a unified view of AWS resources, applications, and services that run on AWS and on-premises servers.

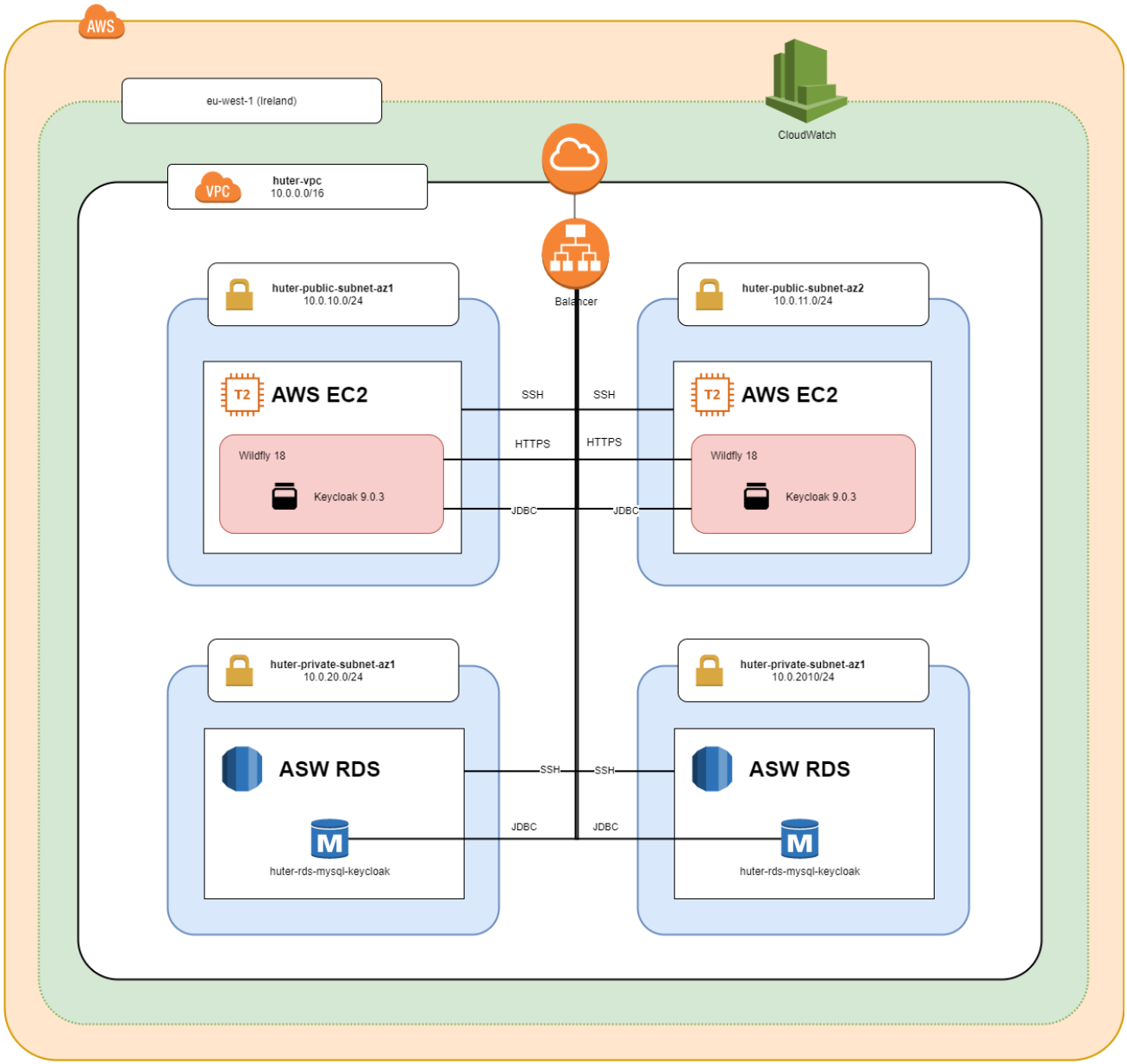
CloudWatch is used for monitoring and management: detection of anomalous behaviour in your environments, setting alarms, logs visualization and metrics side by side, automated actions configuration and issues troubleshooting.

6.2. Operational model

In this section the operational model will be displayed. No further information apart from diagrams will be given because schemas are expected to be clear enough. Additional description will be provided if any further information is considered remarkable.

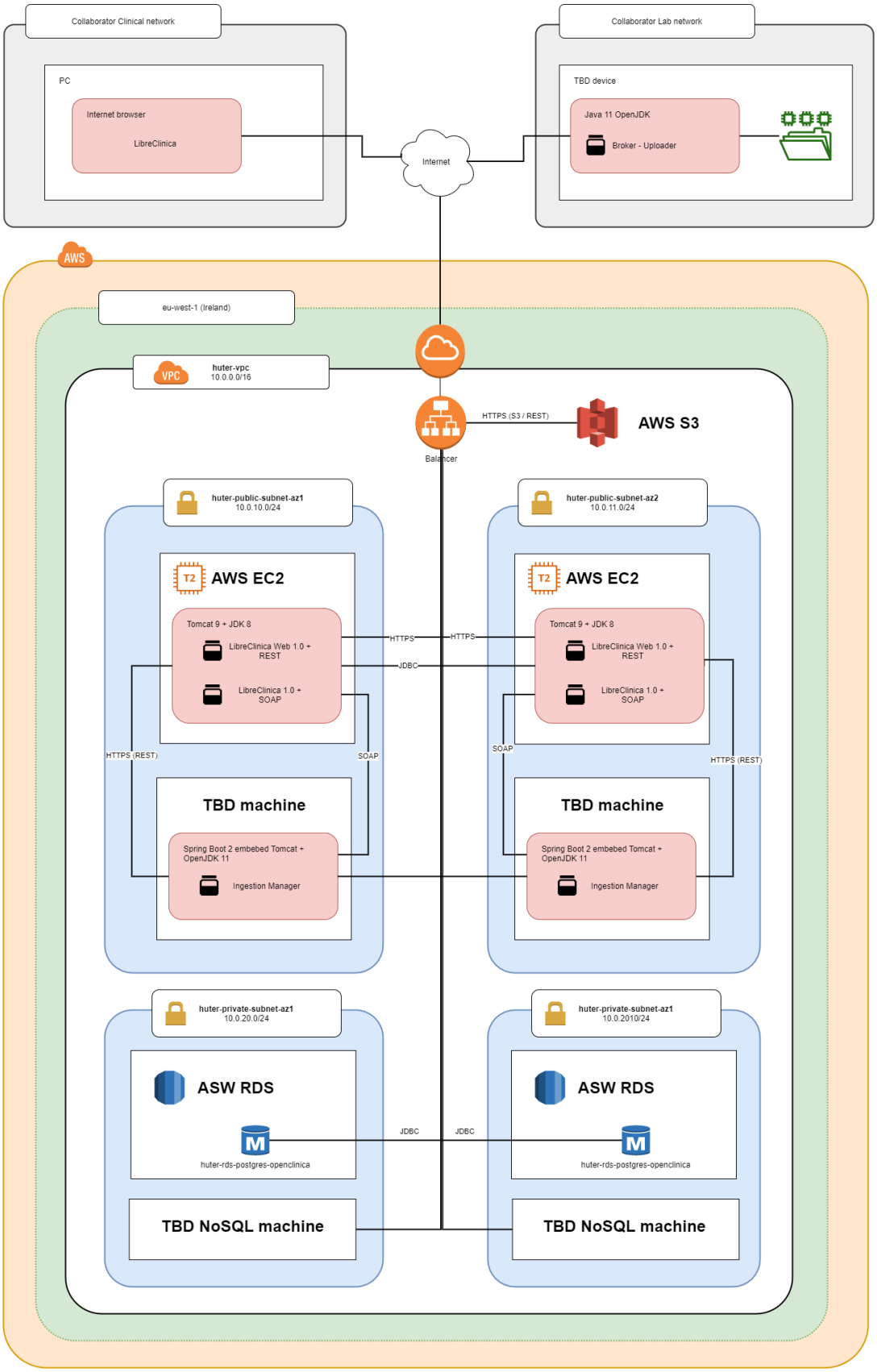
Only current deployed components and totally designed components will be shown in order to prevent confusions regarding the final architecture.

6.2.1. Management deployment model



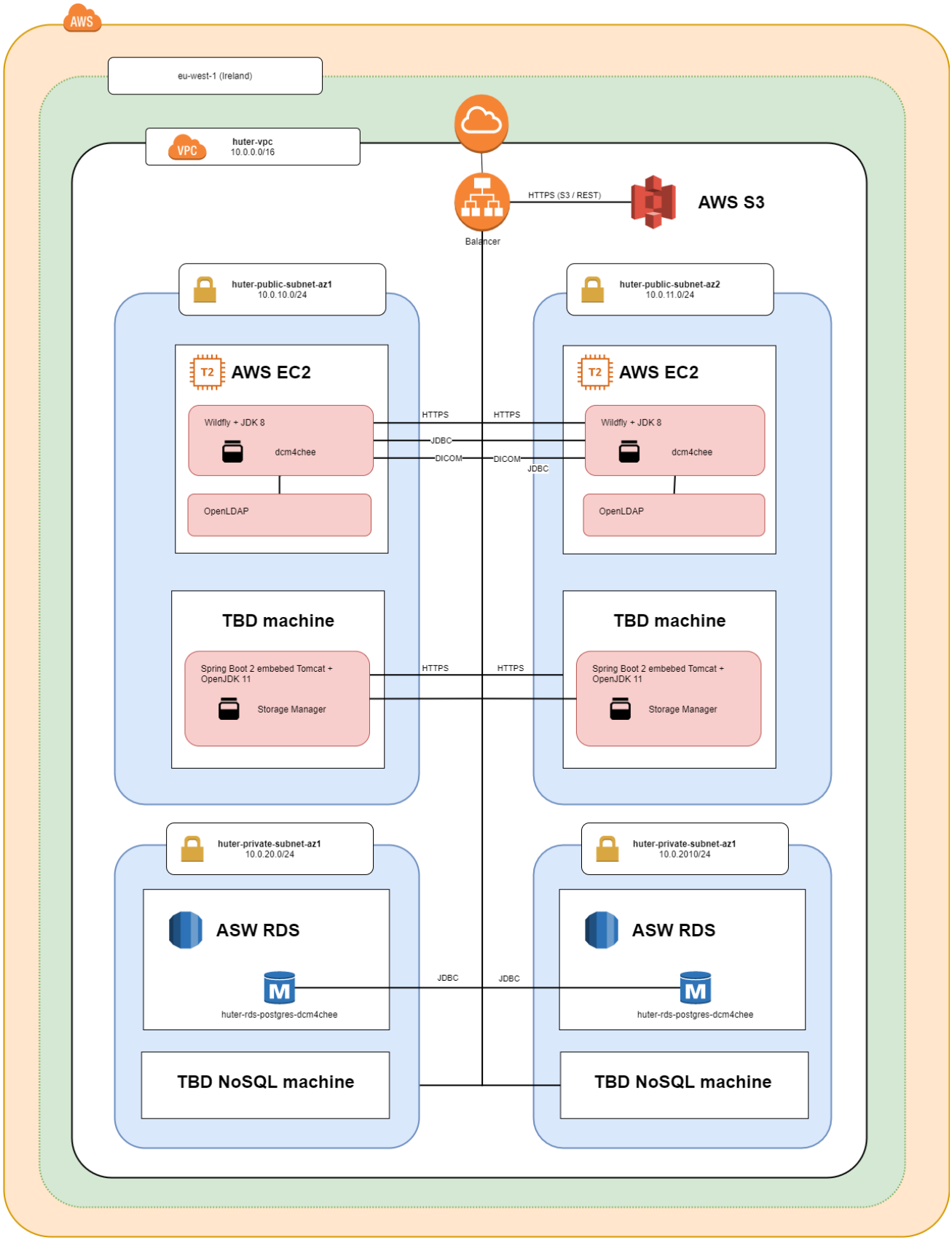
Picture 19 - Management module deployment model

6.2.2. Ingestion deployment model



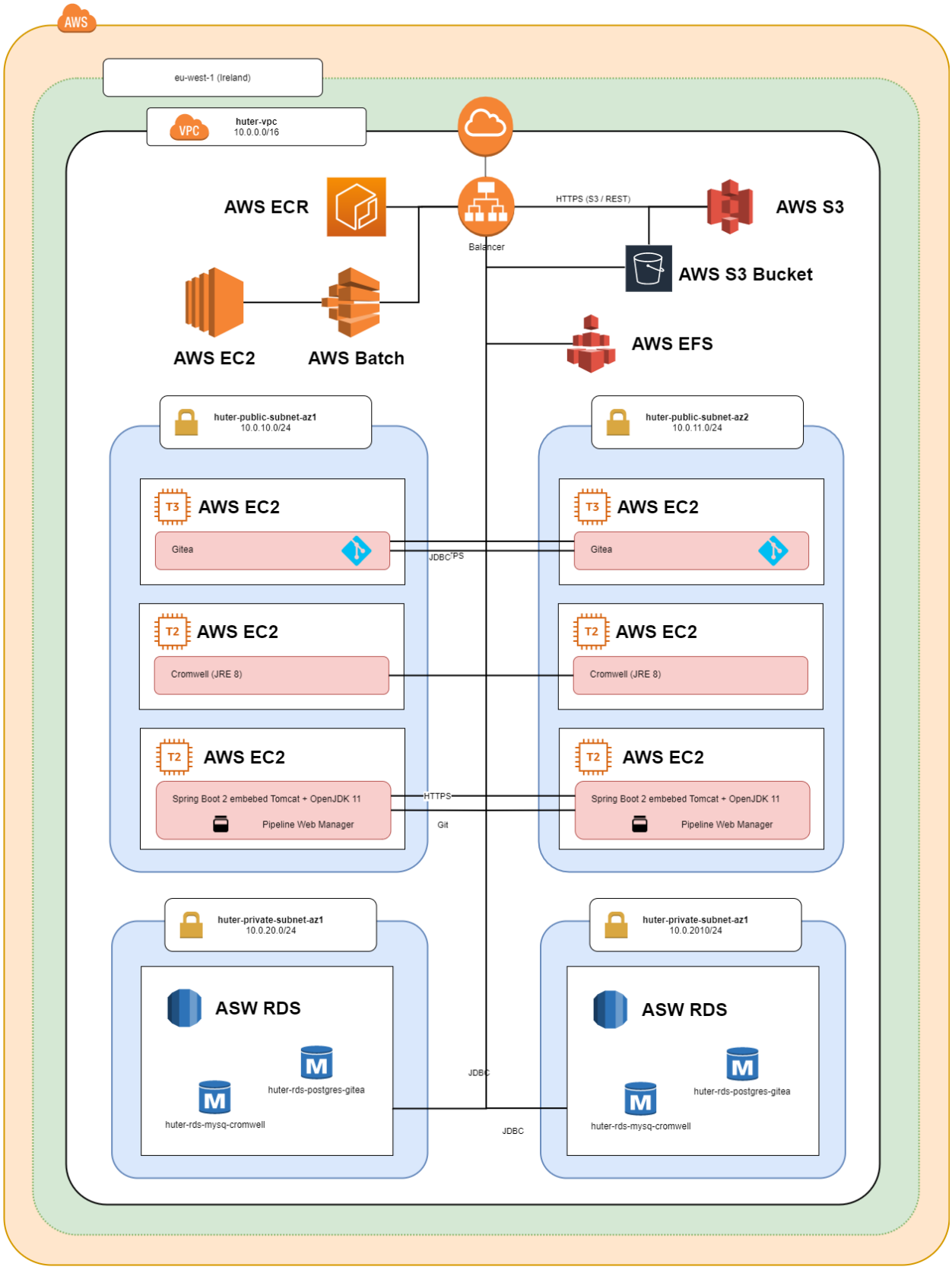
Picture 20 - Ingestion module deployment model

6.2.3. Storage deployment model



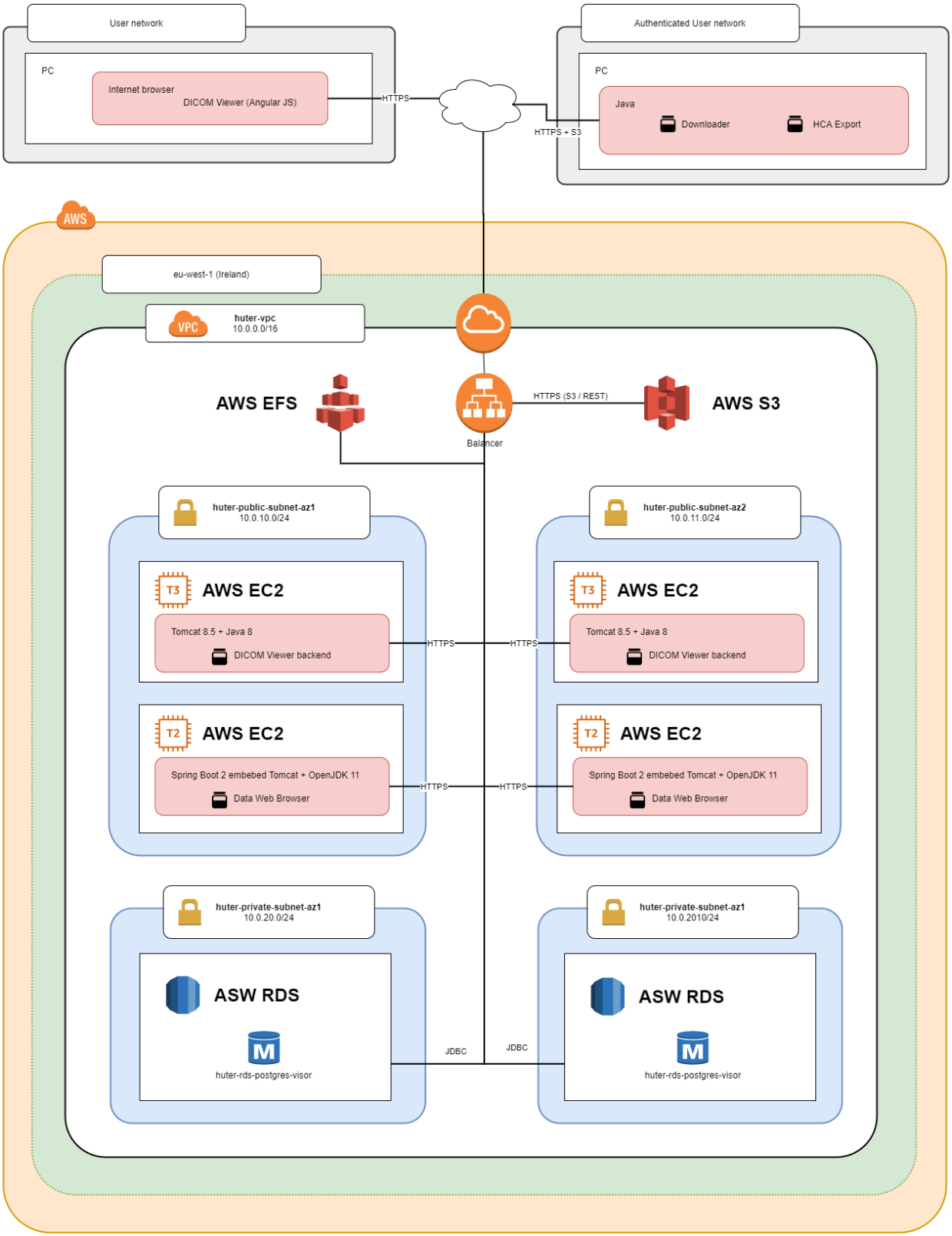
Picture 21 - Storage module deployment model

6.2.4. Processing module deployment model



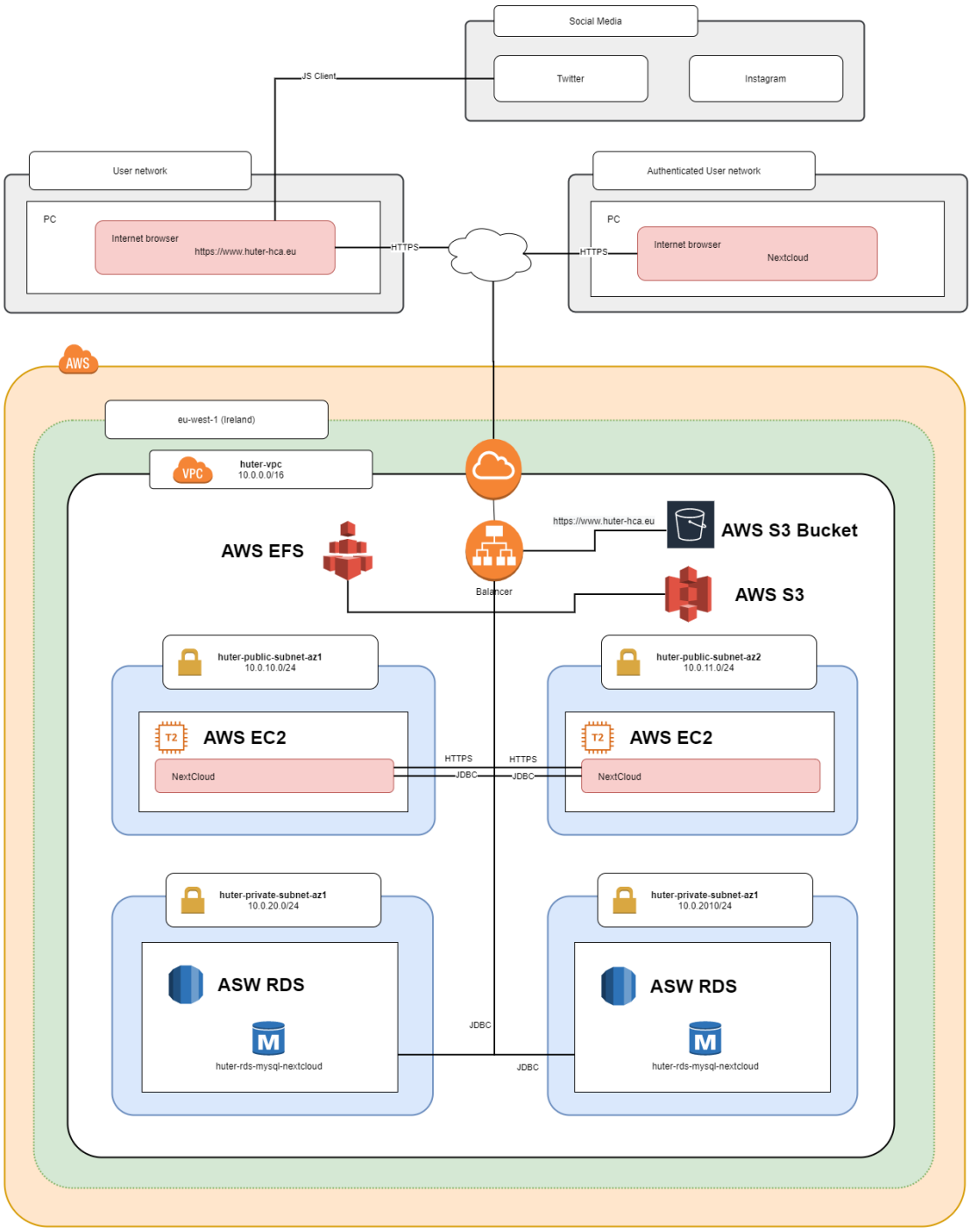
Picture 22 - Processing module deployment model

6.2.5. Data Access module deployment model



Picture 23 - Data Access module deployment model

6.2.6. Dissemination & Communication module deployment model



Picture 24 - Data Access module deployment model

6.3. Technological environment

In order to achieve both functional and non-technical requirements, AWS environment provides all the mechanisms to afford it.

Regarding the ensurance of a stable performance in the HUTER Platform a monitoring process will be executed during the earlier stages. According to the gathered metrics the possibility of changing some AWS services to reach the desirable yield will be evaluated. However, the current infrastructure is able to grow on its own depending on the requested working load so as no further modifications are expected.

6.3.1. Security and access control procedures

Due to the aim of the HUTER project personal and confidential data is stored in the platform. Therefore, a strong security must be applied on all subsystems to ensure data confidentiality across the platform and executed processes.

6.3.1.1. Access control

First of all, a security system to control the access to the platform has been configured. Using Keycloak server and integrating all the application with it, HUTER Platform provides a reliable security based on well-known protocols:

- User authentication is provided using the implementation of OpenID Connect in Keycloak. This feature allows any integrated application to benefit of a Single Sing On system so all applications have to rely on this authentication method.
- User authorization is provided combining security methods to complement OpenId Connect authentication. The first method is the implementation of the OAuth 2.0 protocol in client side because Keycloak already provides the server side functionality based on user roles. However, it is not mandatory to use it so some applications implements their own authorization method typically based on Spring Security due to the nature of the applications technical basements.
- Additionally, and related to the integration of Keycloak, it is important to highlight that the transmission of permissions is done over JWT (JSON Web Token) a reliable and extended method of sharing security information.
- There is an exception to this rule and this is the direct access to the AWS S3 that could be made by using pure AWS credentials.

6.3.1.2. Data security

According to the data security, various layers of protection have been applied to the system. Apart from the access control based on user account and rol managing, the network isolation has been configured. Of course, some entrypoints are open to interact with the application but to use them secure protocols like TLS 1.2 or above have to be applied to the application-protocol. So only HTTPS, S3 over HTTPS and so on are accepted to get in the platform.

When users are allowed to enter to the platform, they will never be able to access to data directly. Only by the interaction through the application can be possible to view or analyze data. If anyone came to access raw data by other forbidden methods, accessed data will be encrypted in order to ensure data integrity. This feature is provided by the S3 storage system applying server side data encryption. AWS Security and Data Protection mechanisms can be reviewed in its website.

On the other hand, a client encryption must be applied when data is sent by client to the platform. This is a guarantee because only HUTER tools will be allowed to send data to the platform. If any other tool is used, communication will be rejected from the server side.

6.3.1.3. Access Log

In order to keep tracking of all the access in the platform and to verify data origins, the HUTER platform will register any action in the platform in the log system. Thus, it will be traceable using Amazon Cloudwatch Tools.

Furthermore, to be compliant with the requirements, it was necessary to apply a data tracking token based on the identifying token on every data registry. This token was defined in the project proposal and it is the subject identifier. Thanks to this token, data can be linked to a specific subject without involving any personal data because it will not be stored anywhere. This approach guarantees subject anonymization as it was defined in the project protocol.

6.3.2. Continuity, capacity and availability

To reach continuity, capacity and availability features, the HUTER platform delegates on AWS Services these responsibilities. AWS has proved a strong agreement with customers building reliable systems and ensuring the ability to answer to any failure. Therefore it can solve any eventuality related to these statements applying next contingency plans.

6.3.2.1. Continuity

AWS replicates applications and data across multiple data centers in the same Region using AZ's. It is possible to choose to increase redundancy and fault tolerance further by replicating data across AWS Regions. This method can be performed using both private, high speed networking and public internet connections to provide an additional layer of business continuity, or low latency access across the globe.

6.3.2.2. Capacity

AWS guarantees the scalability of the configured systems in their platform. Not only it allows the increment of resources machine by machine but also applies automatic horizontal growing and load balancing.

6.3.2.3. Availability

HUTER Platform has been designed to be failure tolerant and be always online. High availability features are based on the load balancing across multiple instances of the same services. Even though not all the services are completely defined at the moment of writing this document, AWS provides mechanism to get rid of a misleading behaviour of an application node instantiating a new one ready to answer any request. AWS can conduct maintenance activities without making any critical service temporarily unavailable to any HUTER collaborator.

AWS obeys Service Legal Agreements listed in here for each of its services. To sum up, regarding AWS S3 availability, AWS ensures at least 95% of system availability.

6.3.3. Operational and system administration procedures

In order to provision the HUTER Platform infrastructure, user behaviour must be taken into account. For instance, it would be important to know the schedule of collaborators to ensure HUTER platform response during that range of time. According to this, HUTER Platform must be full accessible during worktime in Sweden, United Kingdom and Spain. Estonian labs are not expected to use the processing module but data ingest and storage will be accessible during their worktime.

Even when AWS cushions load peaks, the deployment infrastructure has been designed to minimize the activation of this mechanism. The application of microservices approach is an example of this standpoint which tries to reduce automatic or manual system management.

According to this, no additional background processes are expected to be deployed apart from the ones provided by AWS. The main process that could be pointed out is the data replication among AWS S3 datacentres to ensure data security by applying this backup mechanism.

7. 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.

7.1. 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:

7.1.1. 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.

7.1.2. Data querying

This API will provide methods to:

- Query and filter data.
- Download data.

7.1.3. 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.

7.2. HCA alignment

An early study about which data must be sent to the HCA platform was done to facilitate data migration following the HCA guidelines. Technical requirements have also been taken into account to foresee the tools that must be created to export data from the HUTER Platform.

According to the saved data in the HUTER Platform, HCA metadata schemas have been analized to identify the global structure of data and common mandatory fields. Despite the fact HUTER platform will store data side with HCA statements using a self-defined data format, the data will be compliant with HCA ontologies and mandatory fields. Also new variables only related to the HUTER Project will be saved and processed, so their addition to the HCA metadata schemas should not stem the development and exploitation of the HUTER Platform. Besides, the early state of the HCA Platform and legal considerations to use should not stop the HUTER Project progress.

Regarding data format, even when HCA metadata organization is clearly defined, those schemas are alive and will be modified until the moment of the data migration. According to this statement, a final migration process should be needed even if HUTER Platform had integrated HCA data format. In consequence, to reduce the waiting time before the launch of the HUTER Platform, it has been decided to save data using a comfortable format to HUTER researchers. Meanwhile, HUTER teams will collaborate with the HCA Data Wranglers to improve the metadata schemas by requesting the inclusion of HUTER specific variables and ontologies.

As a final step, HUTER data will be migrated using a mapping tool that will help in the format transformation task as well as in the submission to the HCA platform. This process will be agreed with the data wrangler team of the HCA Project to choose the most suitable procedure.

8. 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 ensures subject's right. So anonymization 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.

8.1. General considerations about development. Technology justification.

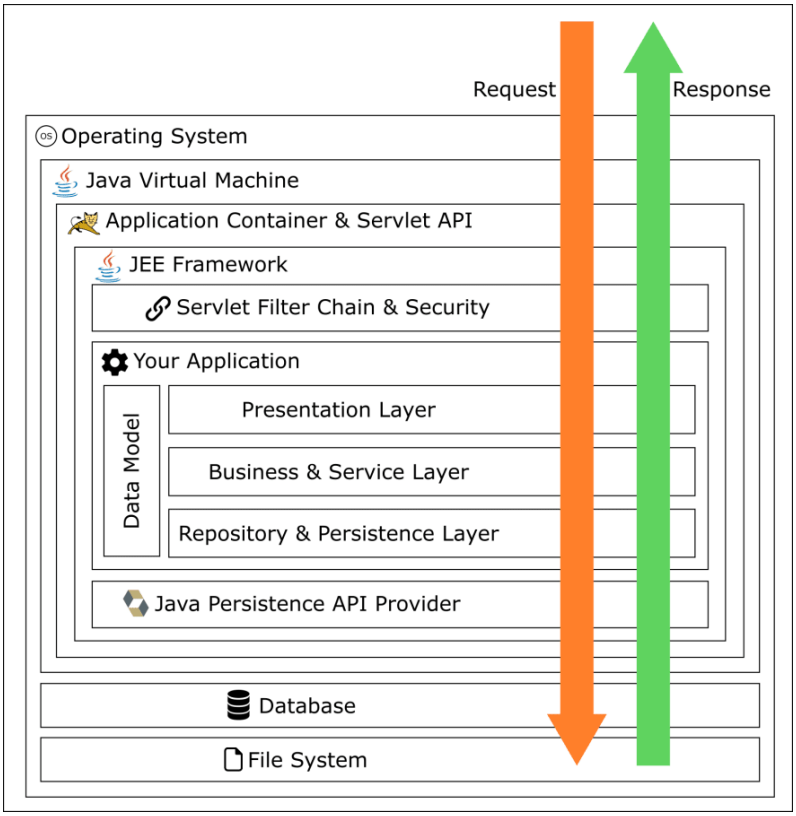
In this section, it will be discussed the decision of the current architecture design as well as the technology environment that was used to build new applications. Regarding existing applications that are deployed in the HUTER Platform, their technology environment was supposed to be no critical. However, their integration layer was crucial in order to be successfully integrated in the HUTER Platform following integration paradigms, so their election was made attending this feature.

In general, most of the applications in the HUTER Platform run or are designed to run in Java. However, it must be pointed out that the complete design has been swayed by the AWS deployment platform. This is the reason why not all the functionalities rely on a Java implementation, taking advantage of AWS services.

In the HUTER Platform, Java was preferred rather than other development language because the maturity of its ecosystem. From development to runtime environment, Java provides a wide set of tools that fits project requirements regardless project dimension. HUTER Project is a large scale Project according to its computing needs so Java allow developers to face this challenge armed with a powerful and flexible language, a varied set of libraries focus on solving different cases of use and wide ranging support tools (IDEs, Application Servers, Continuous Integration/Delivery Servers, Code Reviewers, etc). Besides, Java affords the development using different programming paradigms. The most famous in Java is Object Oriented Programming that will be used in HUTER development (see Picture 25).

So, a good Java advantages summary could be:

1. It is a high-level language. This notion implies a programming language to be much like human language rather than a machine one. Consequently, it should be easy and simple to write, read and maintain.
2. Its stability. The solutions created with the help of Java are said to be stable. That partly happens so because a new version of Java with new features is released every day with advanced features.
3. It is object-oriented. Since Java belongs to object-oriented programming, it allows a developer to write typical programs and to reuse the code. So, one can state classes, generate objects inside classes, work and maintain interaction between two objects.
4. Its maintenance is fairly cheap. The nature of work of a Java program does not rely upon some unique hardware infrastructure, so it is possible to run the server in any machine. Result: it is inexpensive to maintain.
5. It is secure. Java is the first technology which provided security as an integral part of the design. JVM possesses a special identifier which detects the bytecode and checks prior to running.
6. It is multithreaded. Internally a Java program can carry out several tasks at one time.
7. Distributed computing. That is a method when several computers work together in a network. That is definitely an advantage as it allows to develop apps on networks which can contribute to both the functionality of application and data.
8. It is portable (platform-independent). The portability implies that a developer has to write just one code once and the program can be started on any platform. The only condition is that this platform should support JVM.
9. It is robust. Java is said to be a most reliable and powerful language. Its compilers manage to identify every single type of error in your code. Besides that, Java has such great features as exception handling and garbage collection which also prove Java to be reliable.



Picture 25 - Java Layers

According to the software architecture, Java supports the creation of any layer in an application:

- Client side. This is the interface with the user: applets or heavy clients.
- Middle tier. Here is where data gets "formatted" to be sent to the client (JSP, REST API, SOAP).
- Service layer. That is the core of the application where the logic of the application is located.
- Repository layer. It means the information storage in any style: relational databases, NoSQL databases, filesystems, etc.

On the contrary, the huge Java catalogue may make hard to understand the HUTER Architecture if it is explained in such a few words. So, next, java development environments have been grouped to clarify how each application, or group of them, has been developed.

8.2. Technologies

In this section, Java Development Stacks are explained to clarify the building process grouped by developer team or origin.

8.2.1. HUTER Platform

Most of the brand-new software developed to fit HUTER Project needs will be developed using the OpenJDK 11. Java will be used to implement a microservice layer reachable from clients to send or retrieve information as well as lightweight managing applications. These services will be exposed on HTTPS with JSON payload.

8.2.1.1. Design Patterns and approaches

Several approaches will be taken into account in order to minimize the effort of build usable applications in a suitable time.

First pattern to follow is model-View-Controller (MVC). Even though modern web applications are built using the Model-View-View-Model (MVVM), in the HUTER Platform no complex web applications will be provided. So, in order to maximize the delivery frequency, simple and robust MVC applications will be designed.

Furthermore, a microservices approach has been chosen to expose functionality to the client. According to the service development, this paradigm offers:

- Enables the continuous delivery and deployment of large, complex applications.
    - Improved maintainability - each service is relatively small and so is easier to understand and change.
    - Better testability - services are smaller and faster to test.
    - Better deployability - services can be deployed independently.
    - It enables you to organize the development effort around multiple, autonomous teams. Each (so called two pizza) team owns and is responsible for one or more services. Each team can develop, test, deploy and scale their services independently of all of the other teams.
- Each microservice is relatively small:
    - Easier for a developer to understand.
    - The IDE is faster making developers more productive.
    - The application starts faster, which makes developers more productive, and speeds up deployments.
- Improved fault isolation. For example, if there is a memory leak in one service then only that service will be affected. The other services will continue to handle requests. In comparison, one misbehaving component of a monolithic architecture can bring down the entire system.
- Eliminates any long-term commitment to a technology stack. When developing a new service, you can pick a new technology stack. Similarly, when making major changes to an existing service you can rewrite it using a new technology stack.

8.2.1.2. Technological stack

Main technological stack used in Bahia Software developments is based OpenJDK 11.

HUTER Platform requires various applications in order to ingest, store, process and exposed access to data. In regards of this set of applications, each one is designed according to their functionality. However, they share a common technological basement following multi-tier schema.

- Client tier will be built using HTML, CSS and JS tools like JQUery and Vue JS 2 for advance components.
- Middle-tier will provide REST services exposed over HTTPS and using JSON as payload format. Any service will be securized using a Keycloak adapter to integrate OAth 2.0 authorization and OpenID Connect for authentication.
- Service-tier has been designed over Spring Boot 2 and contains the logic of data organization.
- Repository-layer will be based on a NoSQL database, not defined yet. NoSQL approach is the most suitable one for this kind of project according to flexibility about data structure.

Unit testing. Regarding Bahia Software development requirement, unit testing is one the most recommended mechanism to ensure reliable software. So JUnit will be used in order to ensure the expected quality in our results.

Code repositories. HUTER platform is composed of a wide range of application and tools that must be developed, tested, maintained and integrated. Due to this complex system, coordination among developers must be backed with tools to facilitate the development of software. So git has been chosen as the most suitable tools for code versioning because it provides a reliable mechanism on code sharing, merging and modification tracking.

IDE. Software Development Teams use Eclipse as the main IDE because of its integration and wide set of plugins for Java Development. The architecture team works mainly with AWS Console and any transverse tool to leverage the work of the other teams.

Continuous integration. According to Bahia Software development standards, any Java project must run under continuous integration systems implemented in a Jenkins Server. On every code change, Jenkins will perform an automatic build and unit testing execution to check the software health. It also triggers a process in a Sonar instance that performs a code review using predefined Java coding rules and an external knowledge database of bug.

8.2.2. DICOM Viewer

Due to this tool been developed by an external collaborator, they have based its software on a slightly different technological stack.

DICOM Viewer project employs development tools like **Eclipse** as IDE, **GIT** as version-control system, **Gradle** as build automation tool, **NPM** as package manager and **Redmine** as project management and issue tracking tool.

- **Eclipse** is a software platform composed of a set of cross-platform open source programming tools to develop software projects.
- **Git** is an open source version control software, centred in efficiency and reliability of application versioning when they have a large number of source code files.
- **Gradle** is an open source build automation system that introduces a domain-specific language to declare project settings.
- **NPM** is a package management system under Artistic License 2.0.

According to a multi-tier Architecture, applied layers are:

- View layer: Web Interface accessed by external users.
  - It employs AngularJS. It is accessed externally with HTTPS protocol.
- Controller layer: Rest API interface accessed by Web Interface. It joins DICOM Picture Archiving and Communication System (PACS) to retrieve DICOM images. It has a Cache component to prevent duplicate requests.
  - REST API employs Java 8, Spring and Spring Security. It is accessed externally with HTTPS protocol.
  - PACS employs DCM4CHEE. It is accessed internally with HTTP protocol.
- Business layer: It contains business logic and database access.
  - Model layer employs Java 8, Spring and Hibernate.
  - DB employs PostgreSQL 9.5.

8.2.3. LibreClinica

Libreclinica is an Open Source application aimed to manage CRF in a study and it is a fork of a 10 years old application called OpenClinica.

LibreClinica runs over Java 7+, implemented on Spring 5 and Hibernate. It also provides a service layer that mixes REST and SOAP services over HTTPS. Some of the REST services use OpenRosa standard to publish data. According to security layer, it relies on Spring Security for the web portal and WS-Security for SOAP services.

Bahia Software has made an effort to improve this application and adapt some feature to the HUTER project, such as a new method for subject identification. It also has update the front-end replacing old jsp and jmesa with Bootstrap elements. Based on these new requirements, front-end libraries like JQuery have been updated.

8.2.4. Dicomization process

Dicomization process has been designed as a shareable java library that could be integrated in any java process. This library is built with Java 8 and relies in open source libraries (Openslide and Bioformats) in order to read a wide range of digital slides. Then, it transforms these images into DICOM format according to different DICOM Standard, depending on the slide type, using open source dcm4chee library.

8.2.5. Web

To build HUTER website, HUGO framework was used for static pages generation. Look and feel was defined applying Bootstrap 3 as well as pure CSS3 and JS on HTML5.

Further steps will introduce Vue JS 2 in order to add dynamic components in the private area which will be protected using a Javascript adapter of Keycloak.

8.2.6. Deployment environment

In order to develop and test software as fast and safe as possible, a two-environment approach has been applied. Thanks to that, software from development just has to be deployed in DEVELOPMENT (DEV) environment for testing purposes before being ready to use in PRODUCTION (PRD) environment.

With this intention, DEV environment has the same architecture as PRD but providing less processing power.

8.3. Methodologies

HUTER Platform development and support has been divided in three work lines, one of them is being executed by one team:

- HUTER Architecture team: this team is responsible of the deployment infrastructure and supports other teams during the software development.
- DICOM Viewer team: this team is expert in map visualization and is responsible of Dicom Viewer development.
- HUTER Development team: this team develops application to cover required functionalities by the researchers.

Each team is self-organized but during this project it is used a common management model based on the PMBooK standard of Project Management Institute (PMI), and Logical Framework Approach (EML). Although both frameworks have different approaches, they are not exclusive, but can be used in a complementary way to address the management of a project like this one.

EML emphasizes the definition of tasks for the fulfilment of objectives, aligning the final results and the initial approach. For this, an initial analysis of the project is carried out and it is condensed into a Logical Framework Matrix. Although EML makes it possible to define a set of activities while maintaining their coherence with the objectives pursued, it does not satisfactorily address performance monitoring and quality assurance, which PMI does emphasize.

In this project, complex tasks are addressed from their planning, execution and start-up in a possible deployment phase, where it is essential to control and monitor each step, in such a way as to ensure that they are not only the correct execution of the project, but having the necessary elements to verify its viability and sustainability, forces to propose a management, control and monitoring scheme of the project with a methodology that puts the elements and adequate tools to ensure its success.

All these conditioning factors allow proposing a methodological framework for management, control and monitoring based on the integration of EML and PMI. EML is especially relevant in the initial diagnosis of the project for the definition of tasks and PMI to establish a broader framework of action that affects the planning, execution and monitoring of this.

Also, the project incorporates techniques, tools and good practices recommended in the principles of agility (Agile) and, more specifically, in the Scrum framework for application development and Kanban as a method to be applied to agilely manage the needs of specialized support services.

The incorporation of agile methodologies mainly provides the following benefits:

- Increases the ability to respond to change.
- Improves communication and collaboration with the client and end users of the system.
- Allows to focus on complete functionalities and working, with frequent deliveries and maximizing the value provided from the initial phases of the project.
- It facilitates the vision of the development status or support service.