GA-101017441

Personalised Health Monitoring and Decision Support Based on Artificial Intelligence and Holistic Health Records

D2.7 – Functional and Non-Functional Specifications II

WP2 Requirements, State of the Art Analysis and User Scenarios in iHelp

Dissemination Level: Public

Document type: Report

Version: 1.0

Date: August 31, 2022

The project iHelp has received funding from the European Union's Horizon 2020 Programme for research, technological development, and demonstration under grant agreement no 101017441.

Document Details

Project Number	101017441
Project Title	iHelp - Personalised Health Monitoring and Decision Support Based on Artificial Intelligence and Holistic Health Records
Title of deliverable	Functional and Non-Functional Specifications II
Work package	WP2 Requirements, State of the Art Analysis and User Scenarios in iHelp
Due Date	August 31, 2022
Submission Date	August 31, 2022
Start Date of Project	January 1, 2021
Duration of project	36 months
Main Responsible Partner	ENG
Deliverable nature	Report
Authors' names	Claudia Pandolfo, Fabio Melillo, Laura Pucci (ENG), Giorgos Giotis, Spyros Papafragkos (ATC), Diana Kirova (KOD), Pavlos Kranas (LXS), George Manias (UPRC), Oscar Garcia Perales (ICE)
Reviewers' names	George Manias (UPRC), Ainhoa Azqueta, Marta Patiño (UPM)

Document Revision History

Version History			
Version	**Date**	**Author(s)**	**Changes made**
0.0.1	2022-05-15	Claudia Pandolfo (ENG)	Initial Version
0.0.2	2022-06-06	Fabio Melillo, Claudia Pandolfo (ENG)	Infrastructure requirements
0.0.3	2022-06-13	Fabio Melillo, Claudia Pandolfo (ENG)	Non-functional specifications
0.0.4	2022-06-15	Fabio Melillo, Claudia Pandolfo (ENG)	Added data ingestion and data visualisation sequence diagrams

Version History			
Version	**Date**	**Author(s)**	**Changes made**
0.0.5	2022-07-04	Fabio Melillo, Claudia Pandolfo (ENG)	Added sequence diagrams: ▪ Advice follow-up ▪ Advice review ▪ Caption of profiling characteristics ▪ Monitoring ▪ Patient enrolment with DSS and Healthentia ▪ Risk mitigation/ treatment planning
0.0.6	2022-07-11	Fabio Melillo, Claudia Pandolfo (ENG)	Added sequence diagrams: ▪ Advanced risk assessment ▪ Basic risk assessment ▪ Communication of risk ▪ Develop risk prediction model ▪ Elevated risk detected ▪ Plan a visit or control ▪ Request tests and samples ▪ Risk mitigation delivery ▪ Select a preferred risk prediction model ▪ Social network data extraction
0.0.7	2022-07-12	George Manias (UPRC)	Edited section 4.6
0.0.8	2022-07-13	Claudia Pandolfo (ENG)	Edited sections 1, 2, 3.1, 3.2, 3.3 and 5
0.0.9	2022-07-15	Fabio Melillo, Claudia Pandolfo, Laura Pucci (ENG)	Edited sections 3.3 and 3.4, and traceability matrix
0.1.0	2022-07-21	Claudia Pandolfo (ENG), Diana Kirova (KOD), Pavlos Kranas (LXS), Spyros Papafragkos (ATC), Oscar Garcia Perales (ICE)	Addressed comments from technical partners. Ready for internal review
0.1.1	2022-07-22	George Manias (UPRC)	1st Internal Review
0.1.2	2022-07-26	Ainhoa Azqueta, Marta Patiño (UPM)	2nd Internal Review
0.1.3	2022-07-29	Claudia Pandolfo, Laura Pucci (ENG)	Addressed Internal Reviews' comments
0.1.4	2022-08-08	Pavlos Kranas (LXS)	Quality Review
1.0	2022-08-30	Dimosthenis Kyriazis (UPRC)	Final version

Table of Contents

List of Figures

List of Tables

Executive summary

This deliverable presents the second and final release of the functional and non-function specifications of the iHelp solution and updates and replaces the first release D2.6 – "Functional and Non-Functional Specifications I" (Mel, Pan, 21), submitted at M10. Functional and non-functional specifications guide the development activities throughout the project, in order to achieve a consensus on what the iHelp platform will offer. This has a two-fold advantage as it lets the developers know what to build and the users what they will get.

These specifications explain how the requirements provided by the pilots and the technical partners will be fulfilled by the iHelp platform and provide a detailed definition of the functionalities/behaviours of the iHelp components and the interactions among them. Details on the design of the components are outlined in the related deliverables and are out of the scope of this document.

This work started from the initial version of the specifications and took into account the last outcomes of the requirements elicitation and architectural design. More specifically, the reported requirements in D2.3 – "State of the art and requirements analysis III" (M., A., C., + 22) and the updated architecture in D2.5 – "Conceptual model and reference architecture II" (M., P., A., + 22). Moreover, use cases reported in D2.3 have been analyzed to identify new or updated functionalities compared to those considered in D2.6. The final release of the iHelp architecture has been examined to understand the role and objective of each component, and the technical specifications were produced for all iHelp components. What is more, technical partners have provided the Swagger/OpenAPI files documenting the components they are responsible for. Connection mechanisms, data flows and control flows have been discussed and clarified during frequent brainstorming involving all interested parties. Requirements specified in D2.3 have been reviewed to update the non-functional specifications reported in D2.6.

1 Introduction

This is the second and final version of a series of deliverables aiming at describing the functional and non-functional specifications of the iHelp platform, in order to have a common reference of the functionalities offered by the platform, useful for both developers and users of the system. These specifications illustrate how the iHelp platform satisfies the requirements and accurately define the functionalities/behaviours of the iHelp components, thus directing the development activities. Hence, this report focuses on the specification of the necessary assets enabling the components interactions and the usage of the system, on the interfaces of each component, and the communication protocols. Nevertheless, single components and services are here considered as black-boxes and the details of their internal design are reported in the related deliverables and are out of the scope of the present document.

This deliverable represents the final release of the functional and non-functional specifications and updates and replaces the first release D2.6 – "Functional and Non-Functional Specifications I" (Mel, Pan, 21) that focused only on a subset of iHelp components. Due to the incremental development approach adopted by the project, several components, although foreseen in the iHelp architecture, have been designed and implemented after the release of deliverable D2.6 and therefore were not enough mature to be documented at the time.

The present document provides the specifications of the whole iHelp platform, starting from the work presented in the previous version of this deliverable. Thanks to a more detailed knowledge of the components, the functional specifications have been enhanced by providing the Swagger/Open API documentation and sequence diagrams have replaced the activity diagrams included in D2.6 to model the interactions among components.

Other inputs for the specifications were the results of the requirements elicitation and architectural design. The collection of user and technical requirements, provided by the use case partners and the technical contributors of the project, have been included in a series of three deliverables: D2.1 – "State of the art and requirements analysis I" (M., A., C., + 21A), constituting an important input for D2.6, D2.2 – "State of the art and requirements analysis II" (M., A., C., + 21B) and D2.3 – "State of the art and requirements analysis III" (M., A., C., + 22). Being the final deliverable of the series, D2.3 depicts the final user and technical requirements of the iHelp platform. User and technical requirements have been analysed to derive the components constituting the iHelp reference architecture, outlined in its final release in deliverable D2.5 – "Conceptual model and reference architecture II" (M., P., A., + 22). Both D2.3 and D2.5 were useful to gather the specifications described in this document.

This document is organized as follows:

- Section 1 reports the current introduction to the document and details how the content of the first version of this deliverable has been modified.
- Section 2 remarks the iHelp reference architecture and the components being part of it.
- Section 3 describes the API documentation provided as Swagger/Open API files, the interactions among components to fulfil the requirements, modelled with sequence diagrams, and map the requirements to the functionalities provided by the components.
- Section 4 is focused on non-functional specifications.
- Section 5 concludes the document.

1.1 Updated content since D2.6

Section 2 has been revised:

- By adding the Policy Maker and the Admin actors emerged when defining the last version of the requirements, included in deliverable D2.3 (M., A., C., + 22).
- By editing section 2.1 according to the final version of the reference architecture reported in deliverable D2.5 (M., P., A., + 22).
- By updating section 2.1.1, that contains the list of internal components whose interfaces and interactions are documented in the present deliverable. Compared to the previous version of this deliverable, the list of internal components now includes the components not enough mature at the time of releasing D2.6, as well the components added in the final version of the iHelp reference architecture.

Section 3 has been modified:

- By editing section 3.1 related to the API documentation, that is now provided following the Swagger/OpenAPI specifications.
- By updating section 3.2 according to the final list of used cases identified in D2.3 and refined in D2.5. A new use case, namely *Select a preferred risk prediction model*, emerged during the analysis and design of sequence diagram, has been added.
- By completely rewriting section 3.3, as in the present document the interactions among iHelp components have been modelled using sequence diagrams, represent an enhancement of the activity diagrams presented in D2.6.
- By updating the traceability matrix in section 3.4 according to the current version of the API documentation.

Section 4 has been revised to take into account both the updated and newly introduced non-functional requirements reported in deliverable D2.3.

Finally, Annex A was removed, since the API documentation is now available as Swagger/OpenAPI files.

2 iHelp Platform Overview

iHelp aims at early identifying and mitigating the risks associated with Pancreatic Cancer (PC) by applying advance AI-based learning and decision support techniques on the historic (primary) data of Cancer patients, integrated with (secondary) data gathered through the mobile and wearable applications (concerning lifestyle, behavioural, social interactions and response to targeted prevention and intervention measures).

The expected usage of the iHelp platform from a user perspective, detailed in D2.3 – "State of the art and requirements analysis III" (M., A., C., +22), includes:

- Identification of people at high risk of PC and facilitate cancer risk mitigation.
- Delivery and monitoring of targeted recommendations/interventions.
- Evaluation of the risk of toxicity for patients already diagnosed with PC and undergoing radiotherapy or chemoradiation.
- Derivation of the relationships between the known risks and discover new ones if there are new factors that can affect negative or positive modulation.
- Facilitation of the decision-making process into the early diagnosis of those conditions that are elevating the health risk level for PC development, as well as diagnosing the PC in early stage of development.
- Application of data analytics, AI-based algorithms or deep learning technologies to analyse electronic health record data and predict high risk individuals towards pancreatic cancer for early-stage management of the disease.
- Delivery and monitoring of targeted recommendations/interventions.

The actors interacting with the iHelp platform, identified in D2.3 – "State of the art and requirements analysis III" (M., A., C., +22) and described in detail in D2.5 – "Conceptual model and reference architecture II" (M., P., A., +22) are:

- **Health Care Professional (HCP)**: the physicians taking care of the citizens' health and risk of developing the Pancreatic Cancer
- **Individual/Patient**: the citizen/patient enrolled in the iHelp project that will use its functionalities
- **Model Builder**: the data scientist in charge of training the models for better tuning the AI algorithms
- **External Sources**: other external system, such as: laboratory systems (Laboratory), social networks, databases, etc.
- **Policy Maker**: person in charge of making and implementing decision to improve the health and well-being of the public, scanning social media and analysing trends related to eHealth and Pancreatic Cancer.
- **Admin**: technician, such as a Data Manager or Identity Manager, that can interact with the components of the system, also using programmatic interfaces.

2.1 Architecture

The reference architecture of the iHelp platform has been driven by the user and technical (system and software) requirements elicited in D2.3. The iHelp components implementing the requirements were

identified and described in deliverable D2.5 – "Conceptual model and reference architecture II". The final reference architecture of the iHelp platform is shown in Figure 1, where four big blocks are visible:

- The **`Data Storage`** manages the persistence of the data used in the iHelp platform.
- The **`Final user applications`** are the apps dedicated to the human actors of the system (i.e., **`HCP`**, **`Individual`**, **`ModelBuilder`** and **`Policy Makers`**). Their user interfaces will be further described and reported in deliverable D2.9 – "User-Centric Design II", due also in M20.
- The **`Data Ingestion`** includes the components processing primary and secondary data and storing them, compliant with the common data model, in the central the **`Data Storage`**.
- The **`Data Analysis`** and **`Additional features`** blocks abstract from the AI and Monitoring components elaborating the data saved in the **`Data Storage`** to feed the **`Final User Application.`**

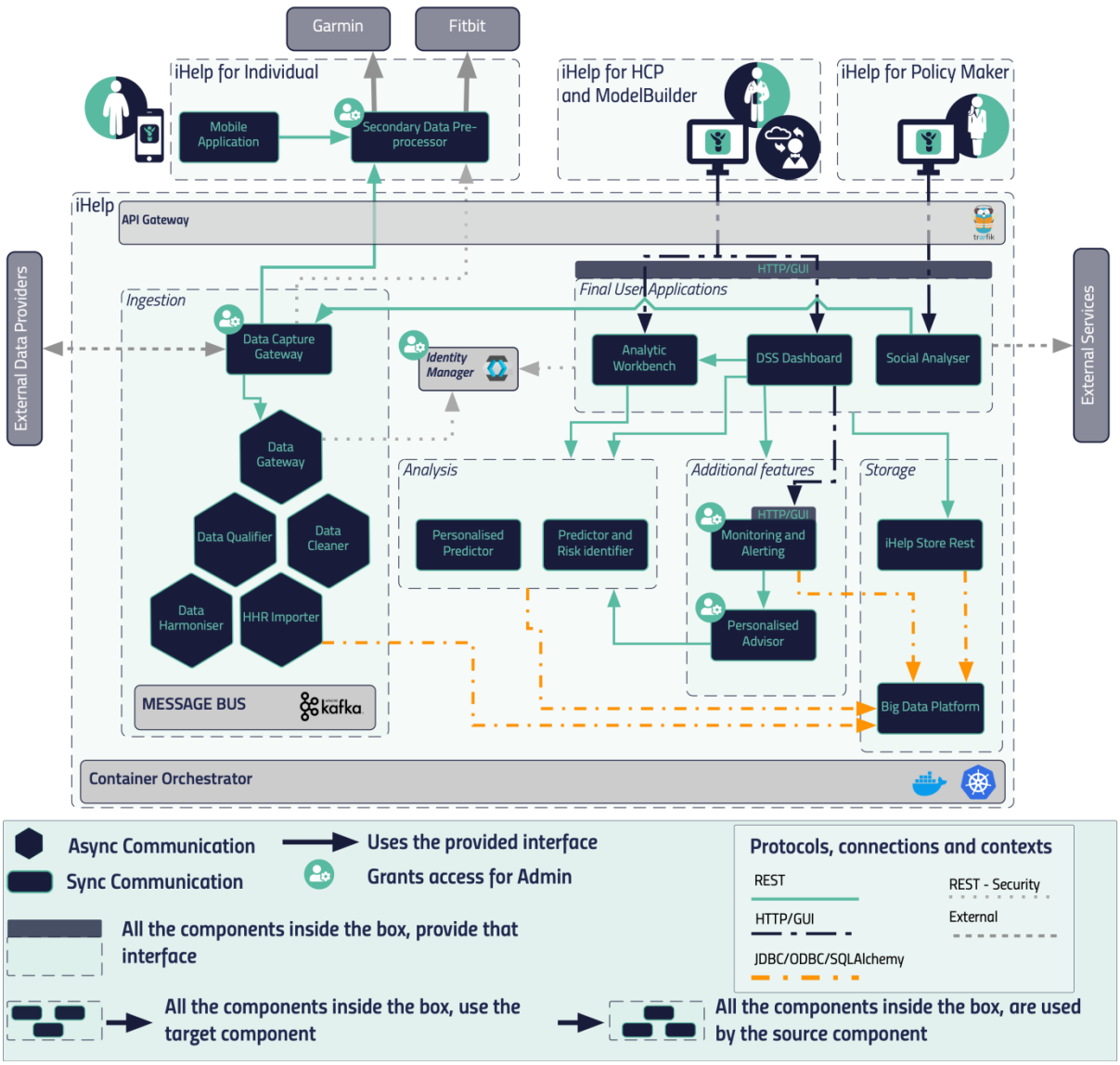

Figure 1: iHelp Architecture (Mel., 22).

The hexagon shape denotes the micro-services that communicate asynchronously, using a message broker (see section 2.1.2), while rectangles represent the components that offer or use an API for asynchronous communication. The type of interfaces used or provided is indicated by the lines connecting the components, as explained in the legend at the bottom of the figure.

The goals and main interactions of the iHelp components are delineated in D2.5. (Mel 22).

2.1.1 Internal components

Specifically, the following list includes the iHelp components, whose external interfaces are documented in the present document, also referencing for each of them the documents where their detailed design are reported:

- Data Capture Gateway, former named Data Connectors (D3.3 (K., M., P. + 21A), D3.4 due at M22, D3.5 (P., S., P. + 21), D3.6 due at M22)
- Data Gateway (D3.3, D3.4)
- Data Cleaner (D3.7 (M., W., D., +21))
- Data Qualifier (D3.7 (M., W., D., +21))
- Data Harmoniser (D3.7 (M., W., D., +21))
- HHR Importer (D3.3, D3.4)
- Secondary Data Pre-processor (D3.5, D3.6)
- Personalised Predictor (planned to be reported in D4.2, due in M22)
- Predictor & Risk Identifier (planned to be reported in D5.2, due in M22)
- Analytic Workbench (D4.4 (GP, L, R, + 22))
- Social Analyser (D5.8 (R, W, G, +22), D5.9)
- Monitoring and Alerting (will be reported in D5.11, due in M22)
- Personalised Advisor (D5.6 (odA, B, P, +22))
- Store REST (planned to be reported in D4.10, due in M22)
- Big Data Platform (D4.9 (K., M., P. + 21B), D4.10)

2.1.2 Components off-the-shelf

The following off-the-shelf components, already available from a commercial source or as open-source software, are part of the iHelp architecture:

- Apache Kafka[1] is an event streaming platform ensuring a continuous flow and interpretation of data. It allows to:
  - Capture data in real-time from event sources like databases, sensors, mobile devices, cloud services, and software applications in the form of streams of events.
  - Store event streams.
  - Manipulate, process, and react to the event streams both in real-time and retrospectively.
  - Route the event streams to different destination technologies as needed. Event streaming thus ensures a continuous flow and interpretation of data so that the right information is at the right place, at the right time.

[1] https://kafka.apache.org/

It is available for all iHelp components, but specifically used for the asynchronous communication of the data ingestion components.

- Keycloak [2] is an open-source Identity and Access Management tool permitting to add authentication to applications and secure services. In the context of iHelp this solution is adopted to allow access only to authorised users. iHelp can be configured to support OpenID Connect (OIDC)[3], an identity layer on top of the OAuth 2.0 protocol that makes also an extensive use of the JSON Web Token (JWT) set of standards.
- Traefik [4] enables routing client API requests to backend microservices and provides key capabilities such as API security, traffic management, and observability. In the iHelp reference architecture it is labelled as API Gateway and will dispatch the external invocation to specific internal services.
- Kubernetes [5] is a portable, extensible, open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation. In iHelp solution this will be useful as orchestrator for the deploy of all the containerised components.

[2] https://www.keycloak.org/
[3] https://openid.net/connect/
[4] https://traefik.io/solutions/api-gateway/
[5] https://kubernetes.io/

3 Functional Specification

This section describes the iHelp platform behaviour, by outlining how the requirements collected in D2.3 – "State of the art and requirements analysis III" (M., A., C., +22) will be addressed by the components being part of the iHelp architecture.

It is intended to be a reference for technical partners and support for development teams to understand the functions of the components and how to interact with them.

These specifications can also help stakeholders to understand whether their requirements have been properly taken into account when designing the architecture and the functionalities of the iHelp components.

3.1 API documentation

This section focuses on the programming interfaces that the iHelp components will provide to interact each other. The API documentation reported in the present document represents an extended or enhanced version of the one included in the previous version of this deliverable, as the knowledge on components has evolved.

Components are here seen as black boxes: functions intended for internal use and implementation details are out of the scope of the present document and will reported in the related WP deliverables (see section 2.1.1).

The API documentation is provided as Swagger, now OpenAPI, files, that can be visualized using the Swagger Editor. The Swagger Editor[6] is an open-source editor to design, define and document RESTful APIs in the Swagger Specification. On the left side of the Editor there is the source code of the swagger file (in yaml or json format) and on the right the HTML page with the documentation generated from such a file.

Swagger files of the iHelp components are available in project's public GitHub repository at https://github.com/ihelp-project/openApi. It should be mentioned that these files will be available for at least 5 years after the projects end.

3.2 Use cases

In the series of deliverables related to requirements, specifically D2.1 – "State of the art and requirements analysis I" (M., A., C., + 21A), D2.2 – "State of the art and requirements analysis II" (M., A., C., + 21B) and D2.3 – "State of the art and requirements analysis III" (M., A., C., + 22), the five pilots have described their purpose and the expected usage of the iHelp platform from a use case perspective, including UML diagrams, and have identified the needs that the architecture should comply with by presenting the list of user requirements. The purpose of these deliverables is to track those requirements throughout the project and update them during the progress of the project; as a result, the third and final version, D2.3, includes the final user and technical requirements of the iHelp platform.

The list of use cases, circulated in D2.3 – "State of the art and requirements analysis III" (M., A., C., + 22) and refined in D2.5 – "Conceptual model and reference architecture II" (M, P, A, +22), has been used to

[6] http://editor.swagger.io

drive the description of iHelp components interaction using sequence diagrams in section 3.3. The new *Select a preferred risk prediction model* use case emerged after the release of D2.5, during the analysis and design of sequence diagrams associated with risk assessment. The use cases taken into account in this deliverable are depicted in Figure 2, where the new use case is highlighted, and listed in Table 1. The use cases marked as NEW have been added after the release of D2.6 (Mel, Pan, 21).

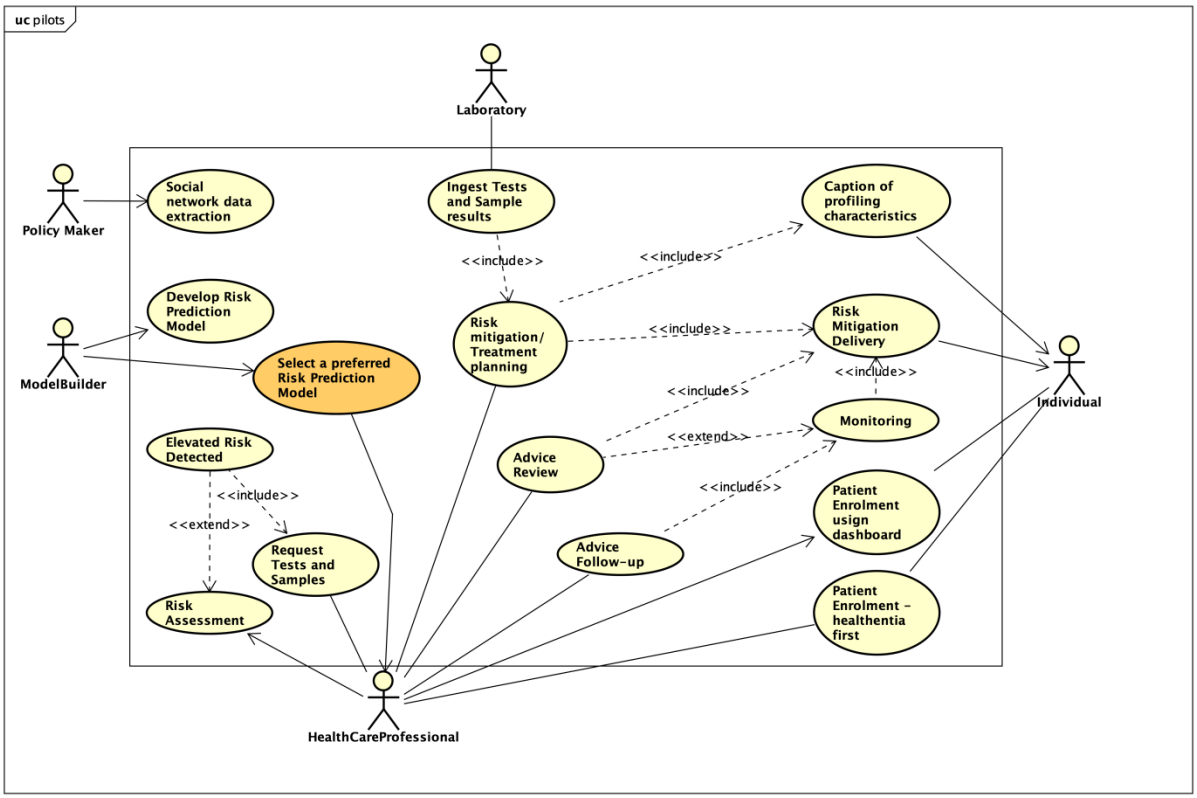

Figure 2: iHelp Use case diagram.

Table 1: iHelp Use cases. The use cases marked as NEW have been added after the release of D2.6 (Mel, Pan, 21).

Use case	Actor(s)	Pilots
Advice follow-up	Health Care Professional Individual	UNIMAN, HDM, MUP, TMU
Advice review	Health Care Professional Individual	UNIMAN, FPG, HDM, MUP, TMU
Caption of profiling characteristics	Health Care Professional Individual	UNIMAN, HDM, MUP, TMU
Communication of risk	Health Care Professional Individual	TMU
Data visualisation	Health Care Professional	FPG, TMU
Develop risk prediction model	Model Builder	UNIMAN, HDM, MUP, TMU
Elevated risk detected	Health Care Professional	UNIMAN, HDM, MUP, TMU
Ingest tests and samples results	External Source (Laboratory)	UNIMAN, HDM, MUP, TMU
Monitoring	Health Care Professional Individual	UNIMAN, FPG, HDM, MUP, TMU

Use case	Actor(s)	Pilots
Patient enrolment using DSS (NEW)	Health Care Professional Individual	UNIMAN, FPG, HDM, MUP, TMU
Patient enrolment using Healthentia (NEW)	Health Care Professional Individual	UNIMAN, FPG, HDM, MUP, TMU
Plan visit/control	Health Care Professional	FPG
Request tests and samples	Health Care Professional	UNIMAN, HDM, MUP, TMU
Risk assessment	Health Care Professional	UNIMAN, HDM, MUP, TMU
Risk mitigation delivery	Health Care Professional Individual	UNIMAN, FPG, HDM, MUP, TMU
Risk mitigation/treatment planning	Health Care Professional Individual	UNIMAN, FPG, HDM, MUP, TMU
Select a preferred risk prediction model (NEW)	Health Care Professional Model Builder	UNIMAN, FPG, HDM, MUP, TMU
Social network data extraction (NEW)	Policy Maker	UNIMAN, FPG, HDM, MUP, TMU

Figure 3 depicts the use cases related to the Admin actors, i.e. the Identity Manager and the Data Manger, identified in deliverables D2.3 – "State of the art and requirements analysis III" (M., A., C., + 22) and D2.5 – "Conceptual model and reference architecture II" (M, P, A, +22), but not detailed due to their technical nature. As mentioned in D2.5, from the analysis of administrative use cases it came out that each of them involves only the interaction between the administrator and one component at a time. As a consequence, admin use cases have been regarded as out of the scope of the present document and the related sequence diagrams have not been produced.

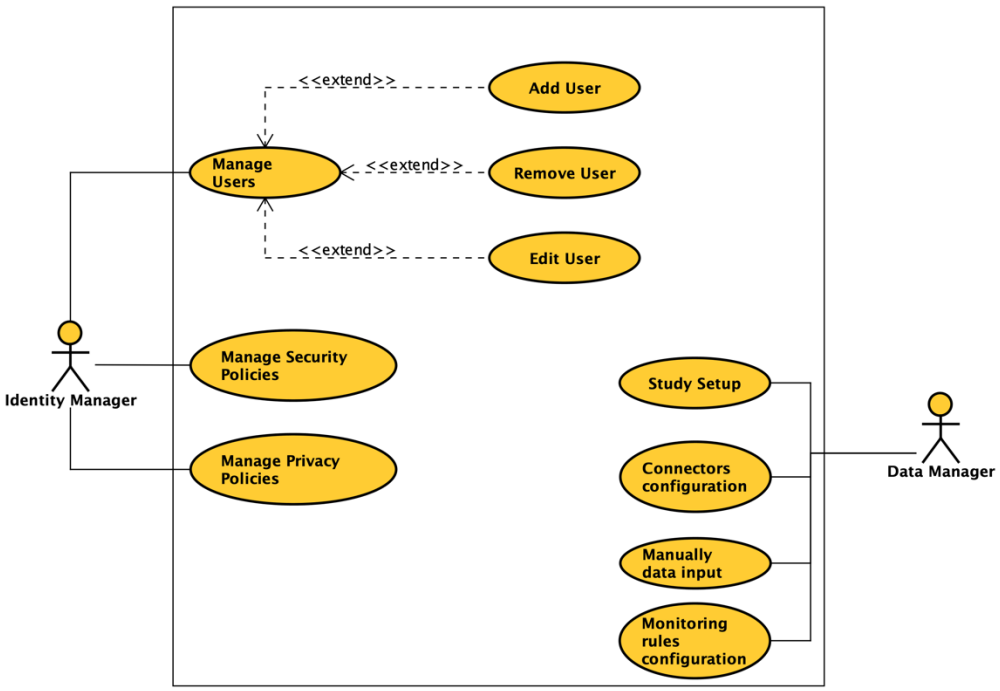

Figure 3: Admin use cases

3.3 iHelp components interactions

Sequence Diagrams model the interaction between different part of a system within a collaboration that realizes a use case. A sequence diagram depicts the components involved and the sequence of messages exchanged between them to carry out the functionality represented by a use case.

The sequence diagrams illustrating how the iHelp components interact to fulfil the use cases listed in section 3.2 are reported in the following subsections, according to the following macroscenarios/topics:

1. Patient enrolment using DSS Dashboard
   Patient enrolment using Healthentia
2. Primary data ingestion
   Secondary data ingestion
   Ingest tests and samples result
   Caption of profiling characteristics
3. Data visualisation
   Risk assessment
   Communication of risk
   Request test and samples
   Plan a visit or control
   Elevated risk detected
   Risk mitigation/treatment planning
   Monitoring
   Advice review
   Risk mitigation delivery
   Advice follow-up
4. Develop risk prediction model
   Select a preferred risk prediction model
5. Social network data extraction

Sequence diagrams represent an enhancement of the activity diagrams presented in the previous version of this deliverable (Mel, Pan, 21) that were designed when information on the exchange of messages among iHelp components was still not sufficient to design sequence diagrams. Activity and sequence diagrams are both behavioural diagrams; the main difference is that the activity diagram represents the flow of activities in a system while the sequence diagram represents the sequence of messages flowing from one component to another.

Sequence diagrams help to visualize the sequence of invocations in a system to perform a specific functionality. The invocations inserted in the sequence diagrams can be:

- The REST APIs included in the Swagger/Open API documentation, when available;
- Conceptual methods if the Swagger/OpenAPI documentation is not available or not needed, as in the case of components providing other interfaces (e.g., graphical or standard, such as JDBC);

- Produce and consume events call for components communicating asynchronously via the Kafka message broker

3.3.1 Patient enrolment using DSS Dashboard

The diagram reported in Figure 4 illustrates the sequence of messages exchanged among iHelp components when the Health Care Professional enrols the patient into the iHelp platform using the DSS Dashboard, by specifying all the available identifiers currently present for the patient.

Three identifiers are stored and used within the iHelp platform:

- The `pilot_patient_ID` is the internal identifier for the patient within the pilot the individual is recruited in
- The `healthentia_patient_ID` is the identifier for the patient used in the Mobile App
- The `iHelp_ID` is the identifier of the patient within the iHelp platform, generated during the registration with the DSS Dashboard

The modelling of these identifiers is done within the context of WP3.

The iHelp components involved in the enrolment at the hospital are the DSS Dashboard, the Store REST and the Big Data Platform.

The sequence is started by the Health Care Professional (HCP), that accesses the DSS Dashboard to register a new patient and is prompted with a registration form. At this point, depending on their knowledge on patient's IDs, the HCP can:

- Enter in the DSS Dashboard the `pilot_patient_ID`
- Enter the `healthentia_patient_ID`, obtained from an eventual previous registration via the Mobile App. If the individual is first being enrolled in the hospital, this identifier is null.

After having inserted the available identifiers for the patient, the Health Care Professional completes the form, the DSS Dashboard internally creates the patient's resource, then stores patient's data in the Big Data Platform via the Store REST component and finally returns to the clinician the `iHelp_ID` generated for the registered patient. If the patient is first enrolled using the DSS Dashboard, when downloading the Mobile App later on he/she will enter that `iHelp_ID` in the app registration form (see section 3.3.2) and will obtain a `healthentia_patient_ID`. At this point, the Health Care Professional will edit patient's data in the DSS Dashboard by adding the new identifier, as illustrated in the *Patient editing with DSS* sequence diagram shown in Figure 5.

The *Patient editing with DSS* sequence diagram, that involves the DSS Dashboard, the Store REST and the Big Data Platform, is started by the Health Care Professional, that accesses the DSS Dashboard to edit patient's data and is prompted with a form filled with the stored patient's data, retrieved by invoking the Big Data Platform via the Store REST component. At this point the HCP can:

- Enter in the DSS Dashboard the `pilot_patient_ID`
- Enter the `healthentia_patient_ID`, obtained from the Mobile App.

When the clinician completes data updating, the DSS Dashboard internally updates the patient's resource, then updates patient's data in the Big Data Platform via the Store REST component.

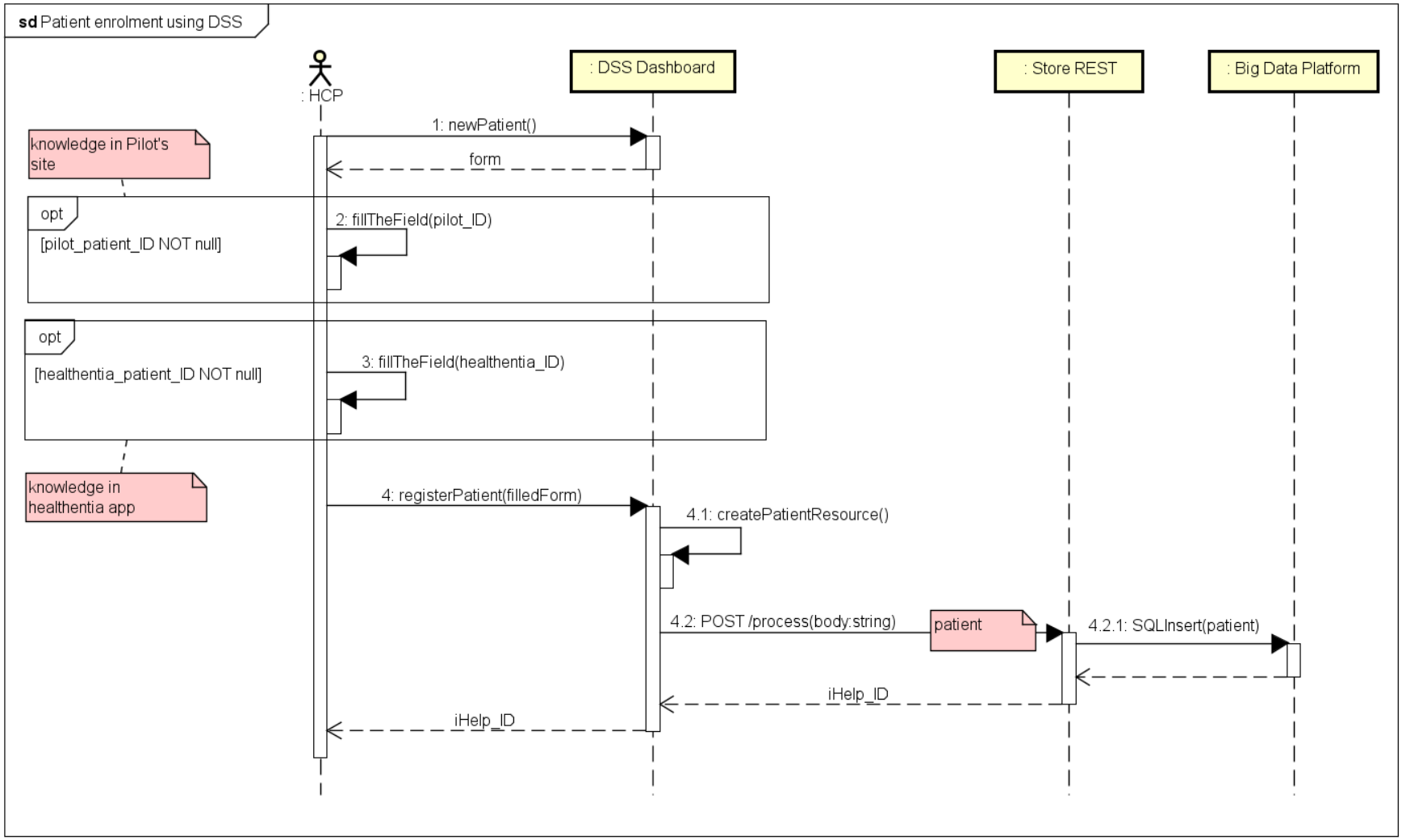

Figure 4: Patient enrolment using DSS Dashboard sequence diagram.

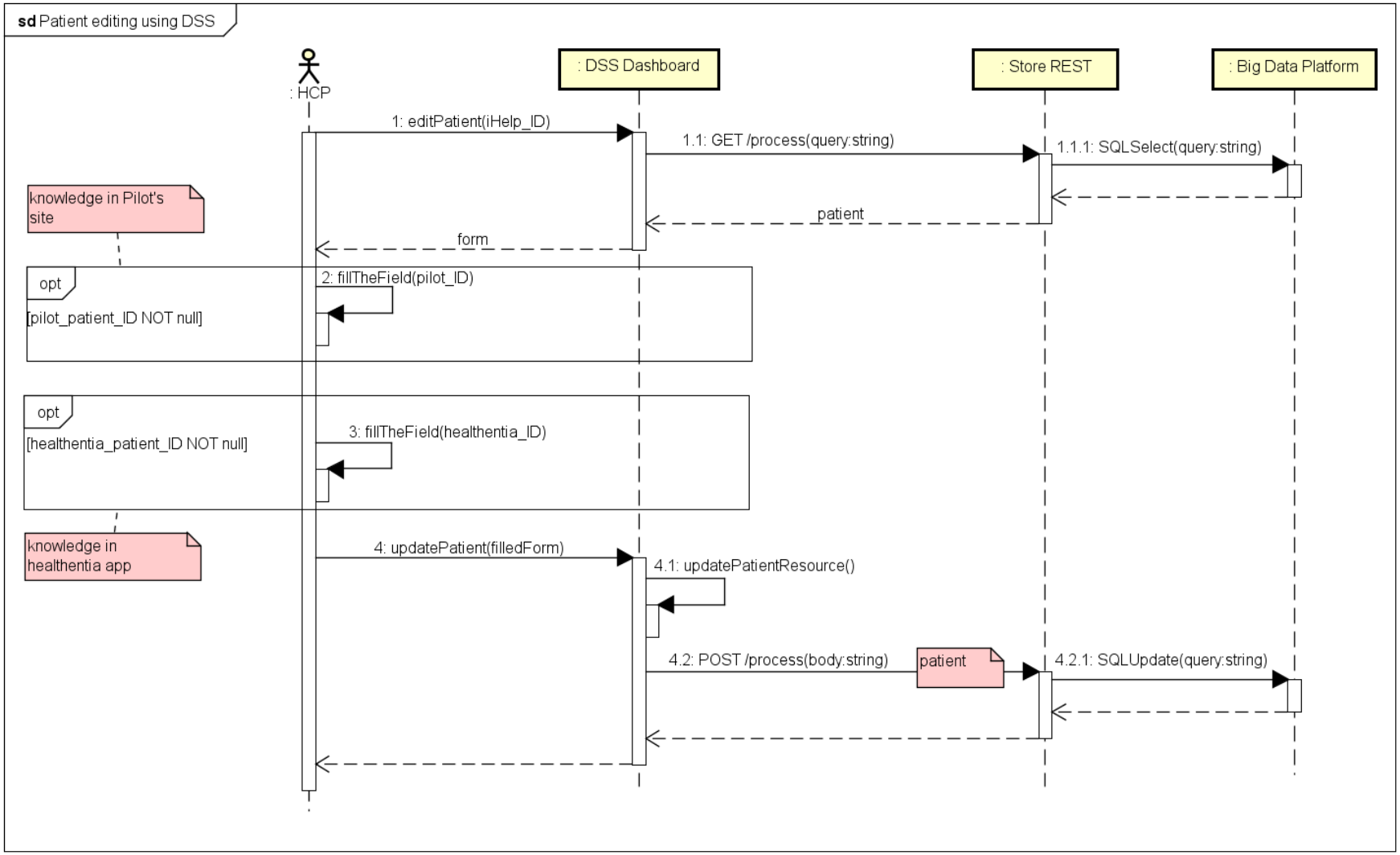

Figure 5: Patient editing using DSS Dashboard sequence diagram.

3.3.2 Patient enrolment using Healthentia

The sequence diagram shown in Figure 6 shows the interactions among iHelp components, in terms of exchanged messages, to allow the Patient to register him/herself into the iHelp platform using the Mobile App, by specifying all the available identifiers currently available for him/her (for further details on the identifiers used in iHelp, see section 3.3.1).

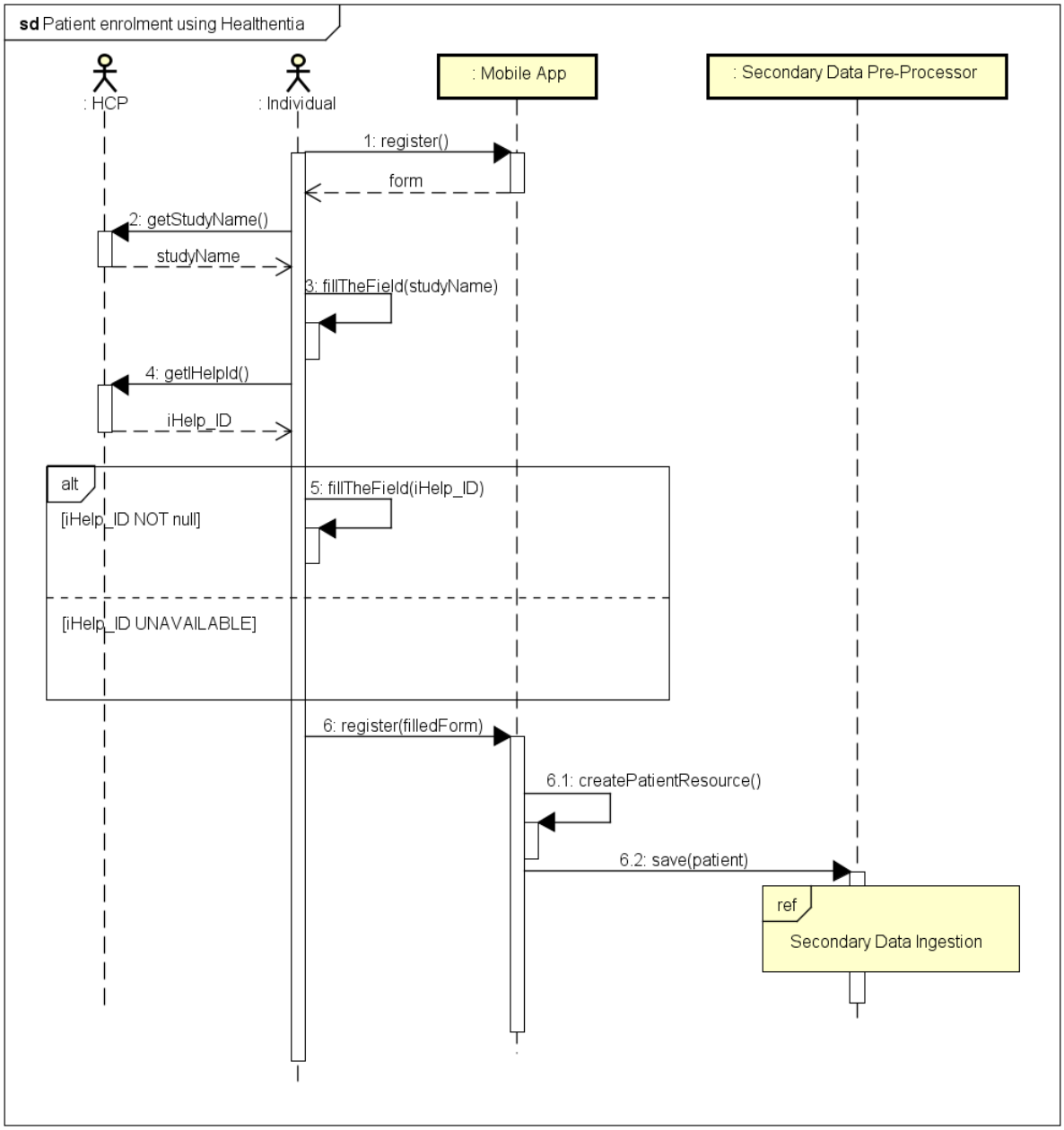

Figure 6: Patient enrolment using Healhentia sequence diagram.

The involved iHelp components are the Mobile App and the Secondary Data Pre-processor.

The sequence is initiated by the individual, that runs the Mobile App and is prompted with the registration form. The required study name, related to the pilot the patient participates to, is received by the clinician and then entered in the form. The `iHelp_ID` is obtained by the HCP if the patient has been

previously enrolled in the hospital using the DSS Dashboard and entered in the form. In case the individual is first being enrolled with the Mobile App, this identifier is null.

Then, the patient completes the registration form. The Mobile App creates the patient resource and pushes new patient's data to the Secondary Data Pre-processor, that starts the secondary data ingestion pipeline described in section 3.3.4.

If the patient is first enrolled using Healthentia, the `iHelp_ID` will be generated at a later time when the clinician registers him/her using the DSS Dashboard. The HCP will then communicate the `iHelp_ID` to the patient, that will enter it in Healthentia, as designed in the *Patient editing using Healthentia* sequence diagram shown in Figure 7.

The *Patient editing using Healthentia* sequence diagram, involving the Mobile App and the Secondary Data Pre-processor, is initiated by the Health Care Professional by communicating the `iHelp_ID` to the individual. The individual starts the Mobile App to edit his/her data and is prompted with a form where the `iHelp_ID` is entered. When data editing is completed by the individual the Mobile App updates the patient resource and pushes updated patient's data to the Secondary Data Pre-processor, that starts the secondary data ingestion pipeline described in section 3.3.4.

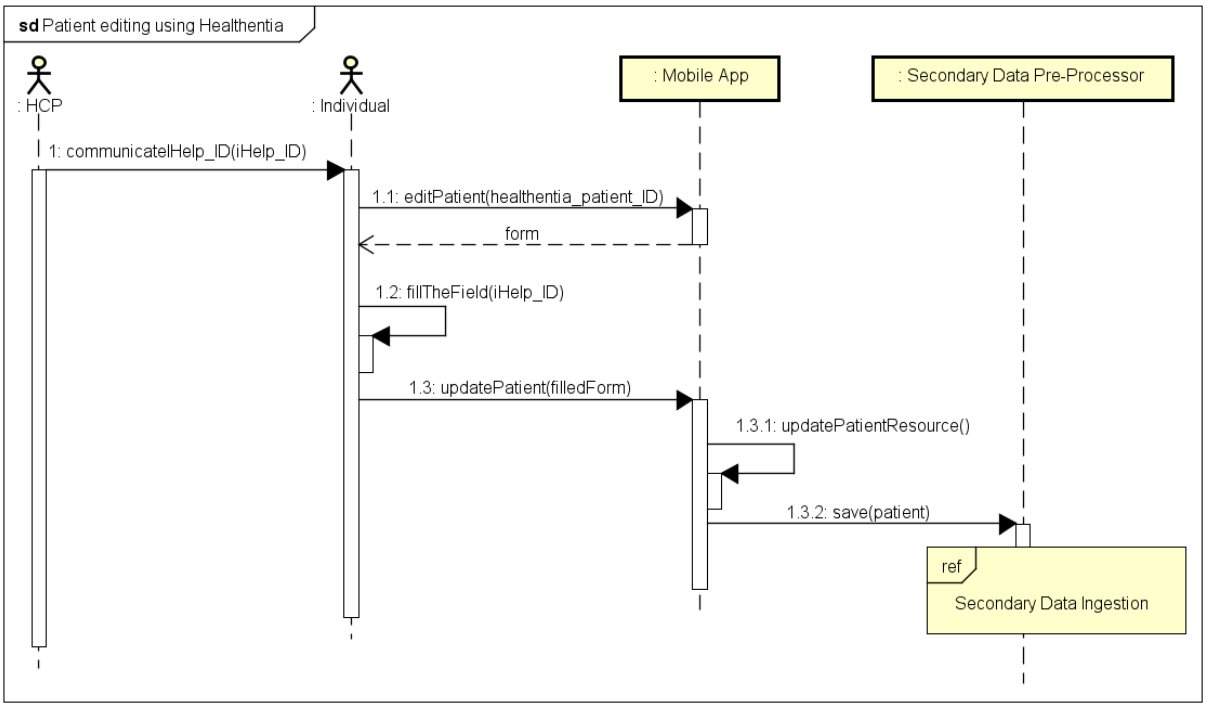

Figure 7: Patient editing using Healthentia sequence diagram.

3.3.3 Primary data ingestion

The sequence diagrams reported in this section describe the messages exchanged by internal and external components when ingesting historic (primary) data of Cancer patients in the iHelp platform. The process of ingesting test and samples in the system is a particular case of primary data ingestion.

To avoid complex diagrams the whole sequence diagram was split in the following three different diagrams.

As shown in Figure 8, the primary data ingestion pipeline is started by the Data Manager, an Admin actor, by invoking the POST `/datacapture(body)` API provided by the Data Capture Gateway to start a data capture job. The request body of this API includes details on the job, especially the `ConnectorArgumentsAbstract.type` , indicating the type of data source to connect to.

Depending on the data source type, three alternative message exchange occur:

- If `ConnectorArgumentsAbstract.type=FTP` the Data Capture Gateway uses the internal FTP Connector to get a static CSV file from an external FTP Server.
- If `ConnectorArgumentsAbstract.type=RelationalDB` the Data Capture Gateway invokes the internal JDBC Connector to execute queries on an external relational database.
- If `ConnectorArgumentsAbstract.type=REST` the Data Capture Gateway calls the internal REST Connector to get data provided by an external server through REST APIs.

Once data from external sources have been captured, the Data Capture Gateway includes them in the request body of the POST `/gateway/publish` API provided by the Data Gateway, by adding also the `api_keys` in the head of the request. The Data Gateway checks the provided security keys by calling Keycloak via `checkSecurityKeys(api_keys)`. If the component is not authorised a 401 error is returned, and the primary data ingestion terminates. If the provided security keys are valid, the Data Gateway internally transforms the message into an appropriate format and sends it to the message broker with topic T_1 via `produce(event,`T_1`)`.

The components involved in the following step of the primary data ingestion, namely Data Cleaner, Data Qualifier and Data Harmonizer, have previously subscribed to interested Kafka topics using `subscribe(topic)` method, as illustrated in Figure 9.

Once the Data Cleaner is notified by the message broker of an interested event with the `consume(event,`T_1`)` call, it validates (with `validateData(data)` method taking data included in the event notified by Kafka), cleans (`cleanData(data)` method) and verifies (`verifyData(data)` method) the received data. At the end of this process, the Data Cleaner sends the clean data to the message broker with topic T_2 via `produce(event,`T_2`)`.

When the message broker invokes the Data Qualifier with the `consume(event,`T_2`)` call, it evaluates the quality of the data embodied in the event using the internal `retrieveDataQuality` method. If the retrieved quality is acceptable according to predefined thresholds, the Data Qualifier forwards data to the message broker with topic T_3 via `produce(event,`T_3`)`.

As soon as the Data Harmonizer is notified of an interested event with the `consume(event,`T_3`)` call from the message broker, it harmonises (with `harmoniseData(data)` method taking data embodied in the event notified by Kafka) and annotates (`annotateData(data)` method) the received data and then invokes the `mapToHHR(data)` method provided by the Primary Mapper to map the annotated data to the common HHR format. At the end of this primary ingestion step, the Data Harmonizer sends the generated HHR to the message broker with topic T_4 via `produce(event,`T_4`)`.

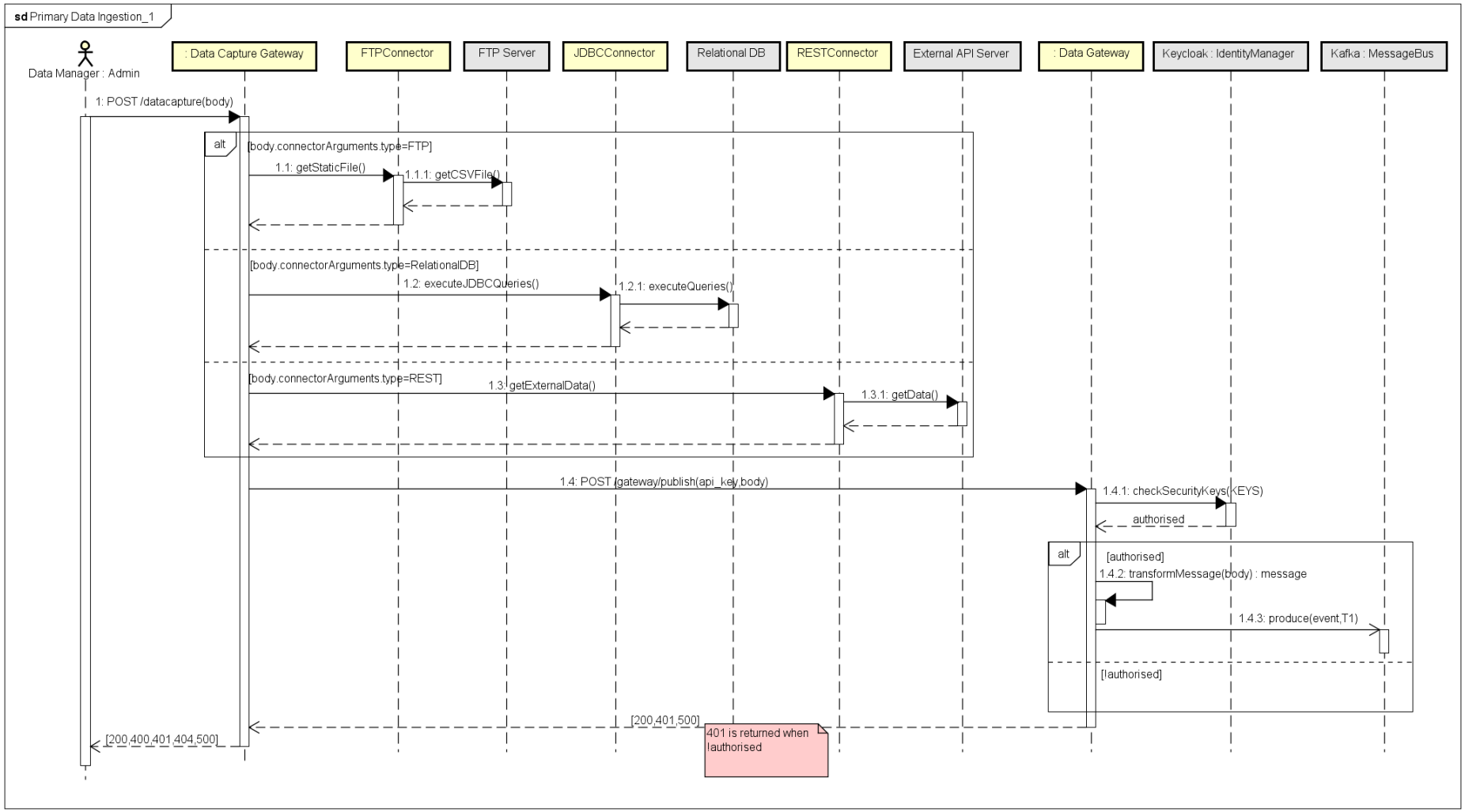

Figure 8: Primary data ingestion – step 1 sequence diagram.

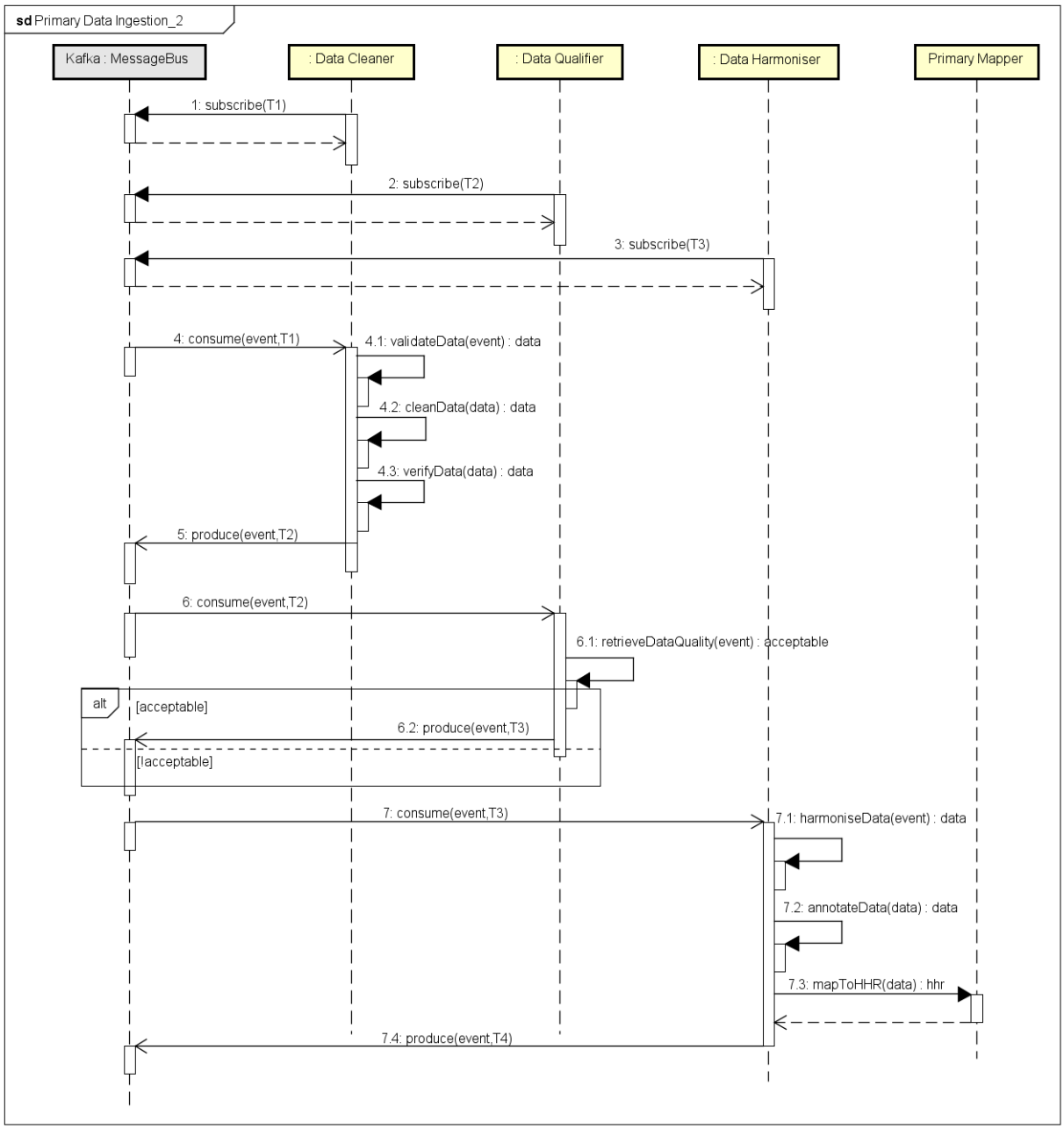

Figure 9: Primary data ingestion – step 2 sequence diagram.

In the final step of the primary data ingestion pipeline, shown in Figure 10, the HHR Importer, previously subscribed to topic T_4 using `subscribe(`T_4`)`, receives the HHR when is invoked by the message broker with the `consume(event,`T_4`)` call and transforms with `transformHHREntities(event)` the HHR entities in tables suitable to be stored in the Big Data Platform through the method `storeHHR(transformedEvent)`.

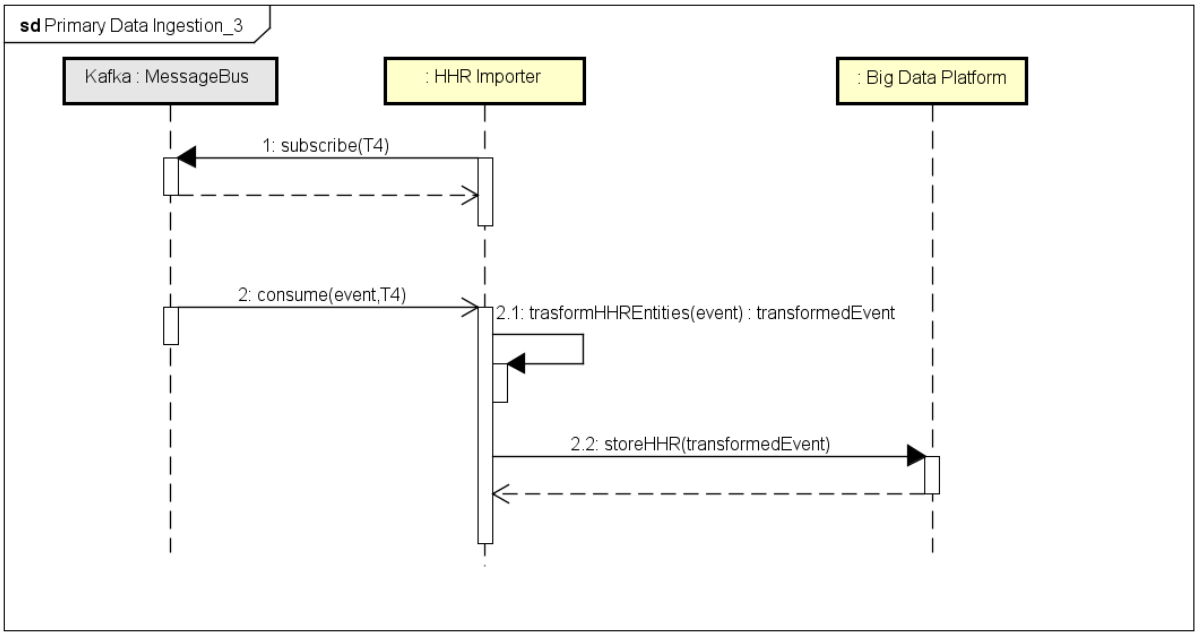

Figure 10: Primary data ingestion – step 3 sequence diagram.

3.3.4 Secondary data ingestion

This section describes the sequence diagrams designing the interactions among internal and external components to ingest secondary data, gathered through the mobile and wearable applications, in the iHelp platform.

The secondary data ingestion pipeline is also used to change the status of messages sent to patients via the Mobile App to:

- Communicate the estimated risk (see section 3.3.9).
- Mitigate risk (see section 3.3.16).
- Plan visits and controls (see section 3.3.11).
- Request test and samples (see section 3.3.10).

To improve readability, the overall sequence diagram has been separated into three different diagrams.

The mobile app pushes data to the secondary data pre-processor, that collects data also from external servers and tries to extract higher level info from the raw data. All data, raw and higher-order are made available to authorised systems via the Secondary Data Pre processor API invoked by the Data Capture Gateway to get the data it needs. In particular, the Data Capture Gateway:

1. Uses the `POST /account/Login` endpoint to authenticate as “application”, whose credentials can be used to access only the API, and get a token (and refresh it if necessary with the `POST /account/RefreshToken` endpoint). All Secondary Data Pre processor endpoints need the token.
2. Uses the data provider endpoints to get questionnaire answers (GET /data-provider/answers), physiological data (GET /data-provider/physiological) and exercise sessions (GET /data-provider/exercises) about a particular patient.

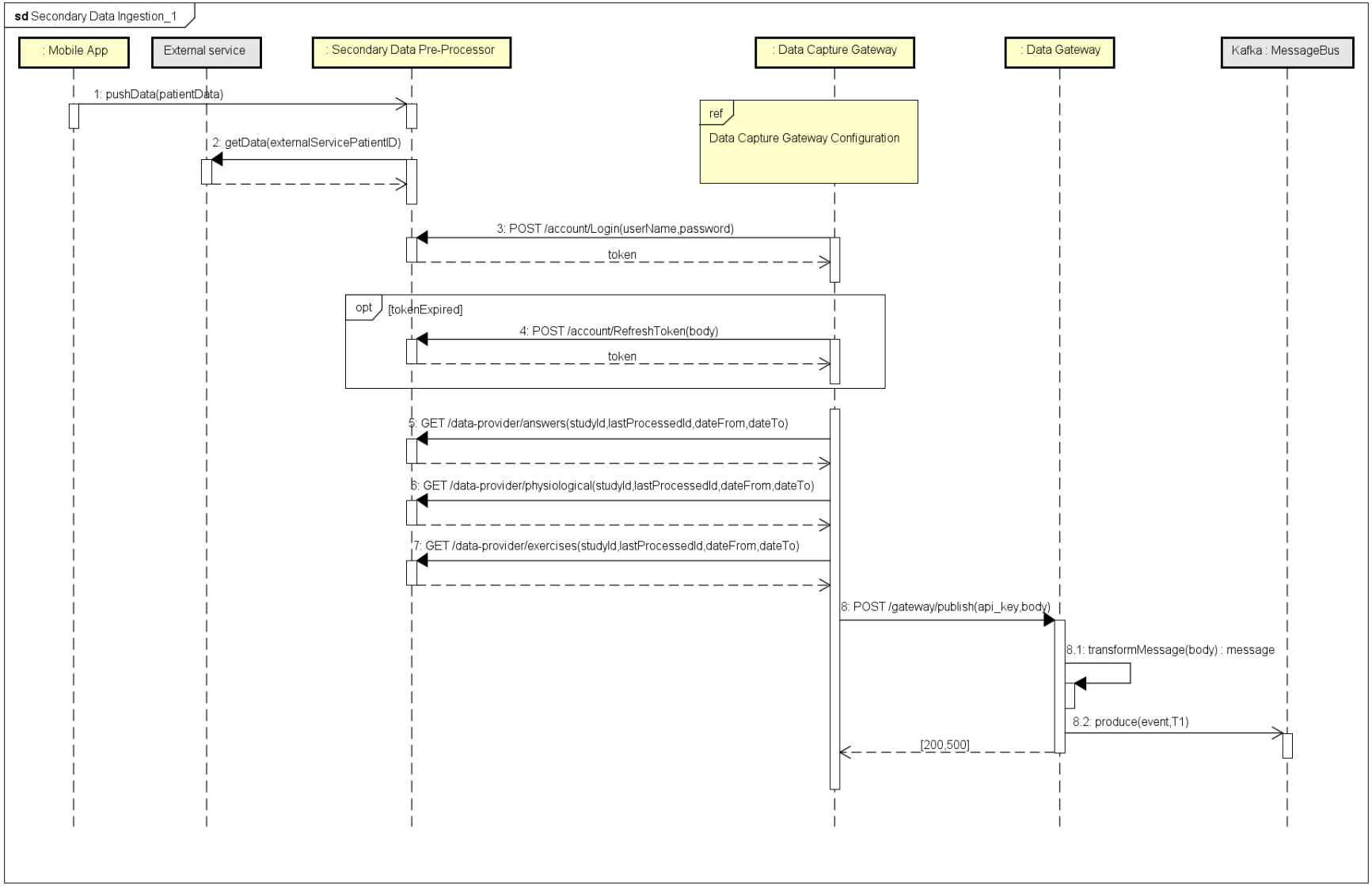

Figure 11: Secondary data ingestion – step 1 sequence diagram.

Similarly to what occurred during the primary data ingestion, once secondary data have been captured, the Data Capture Gateway includes them in the request body of the `POST /gateway/publish` API provided by the Data Gateway. Check of credentials is not needed in this case because the Data Capture Gateway can be already regarded as "trusted" after being authenticated in the iSPRINT server, and the Data Gateway internally transforms the message into an appropriate format and sends it to the message broker with topic T_1 via `produce(event,`T_1`)`.

The message exchange modelled in the second step of the secondary data ingestion (see Figure 12), is rather equal to the one designed in the sequence diagram shown in Figure 9, with the exception that the Data Harmonizer calls the Secondary Mapper component to map annotated data to the common HHR format.

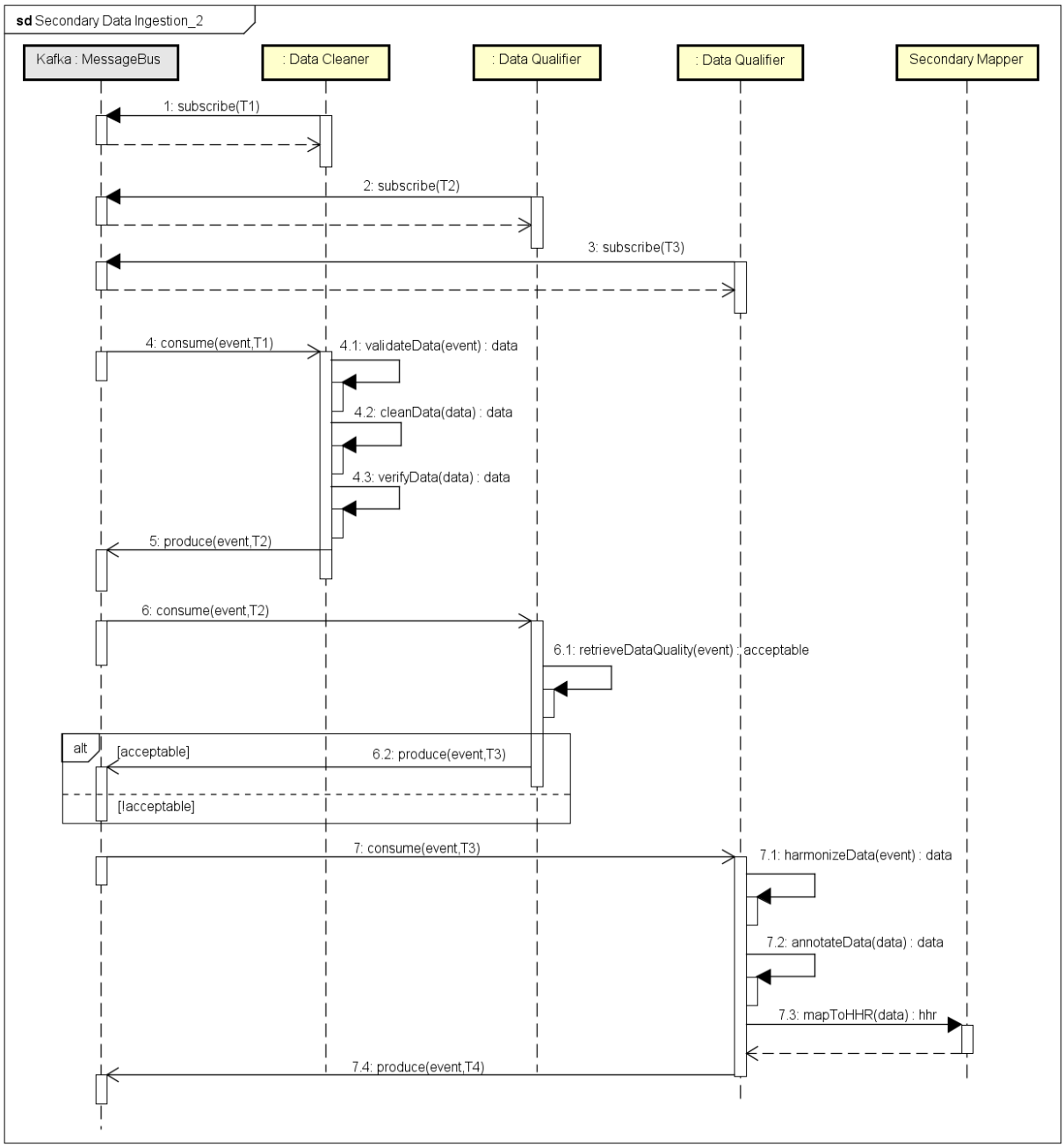

Figure 12: Secondary data ingestion – step 2 sequence diagram.

In the third and final step of the secondary data ingestion pipeline, shown in Figure 13, equal to the final step of the primary data ingestion pipeline modelled in Figure 10, the HHR Importer stores the HHR entities tables in the Big Data Platform, as already described in detail in section 3.3.14.

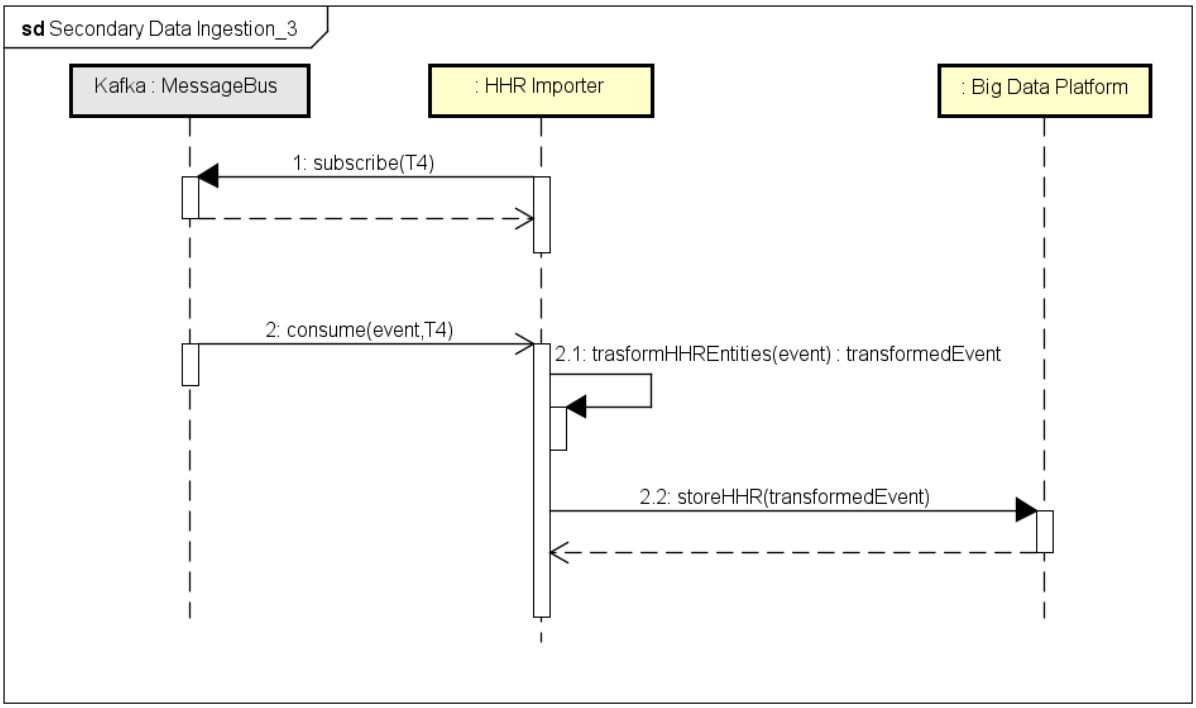

Figure 13: Secondary data ingestion – step 3 sequence diagram.

3.3.5 Ingest tests and samples result

This use case deals with the activities to be performed to ingest data in the iHelp platform. The data ingestion pipeline processes and stores data derived from different sources, also external to the iHelp project, both historic (primary) data of Cancer patients (see section 3.3.3) and (secondary) data gathered through the mobile and wearable applications (see section 3.3.4). The process of ingesting test and samples in the system is a particular case of primary data ingestion.

3.3.6 Caption of profiling characteristics

The sequence diagram displayed in Figure 14 describes the sequence of messages exchanged among the Secondary Data Pre-Processor, the Data Capture Gateway, and the Data Gateway when profiling individuals involved in the pilots.

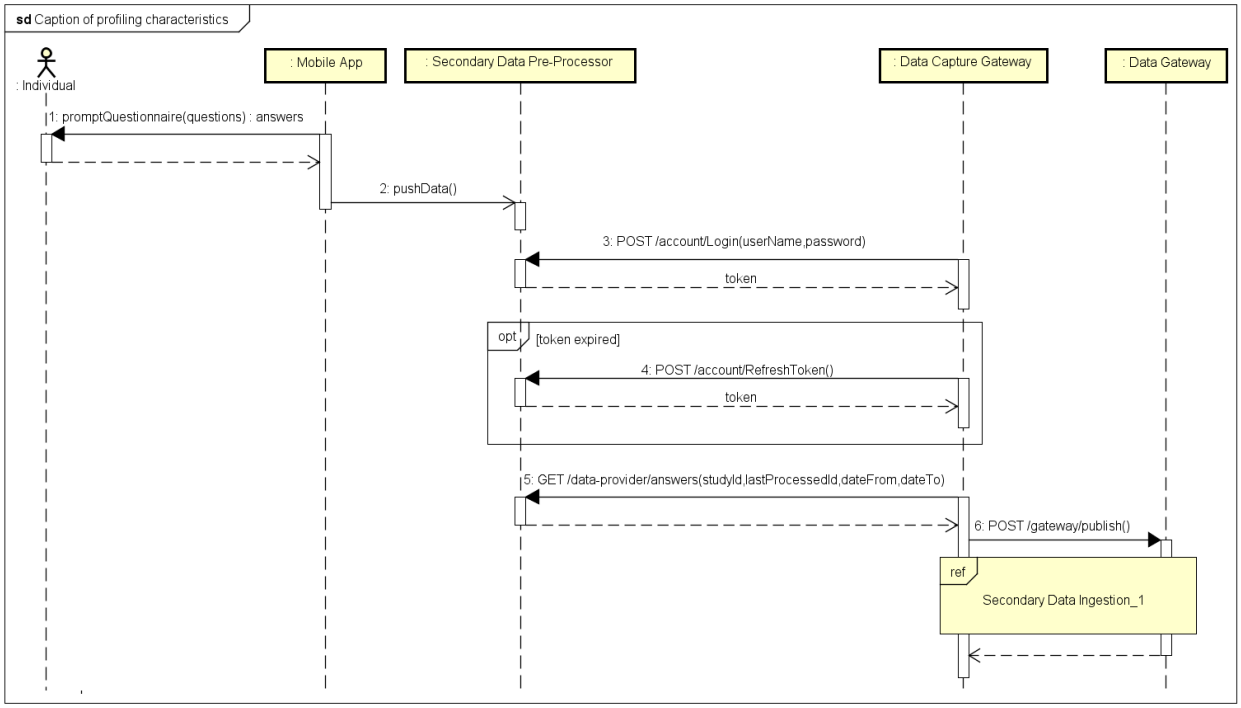

Figure 14: Caption of profiling characteristics sequence diagram.

The sequence is triggered by the Individual engaged in the pilot that, after having launched the Mobile App, is prompted with a questionnaire and provides the required answers. The mobile app pushes these answers to the secondary data pre-processor, that makes all data available to authorised systems via the Secondary Data Pre processor API, invoked by the Data Capture Gateway to get the data it needs. In particular, the Data Capture Gateway:

1. Uses the `POST /account/Login` endpoint to authenticate as "application", whose credentials can be used to access only the API, and get a token (and refresh it if necessary with the `POST /account/RefreshToken` endpoint). All Secondary Data Pre processor endpoints need the token.
2. Invokes the GET /data-provider/answers endpoint to get questionnaire answers.
3. The Data Capture Gateway includes data related to questionnaire answers in the request body of the POST /gateway/publish API provided by the Data Gateway.

The messages exchanged from this point forward are modelled in the *Secondary data ingestion* diagrams (see section 3.3.4).

3.3.7 Data visualisation

The sequence diagram depicted in Figure 15 illustrates the sequence of messages exchanged among the DSS Dashboard, the Store REST, the Big Data Platform and the external component Keycloak to carry out the *Data visualisation* functionality, allowing the clinician to view and analyse collected patient data.

This diagram includes the calls required to authenticate and authorise the Health Care Professional accessing the DSS Dashboard and is therefore referenced by other sequence diagrams.

The sequence is initiated by the Health Care Professional actor that tries to access the DSS Dashboard with the Login. Once received the `Login(iHelpID,password)`, the DSS Dashboard invokes Keycloak with `checkCredentials(id,password)`, that returns a boolean indicating whether the user is authorised or not. If yes, the DSS Dashboards accesses the Big Data Platform, via the GET `/process(query)` endpoint provided by the Store REST component, to obtain the authorised Health Care Professional data, including their patients. The Health Care Professional then selects the iHelp_ID of a specific patient and the DSS Dashboard calls twice the Store REST component using GET `/process(query)` to retrieve the primary and secondary data of the patient and visualises them in a report.

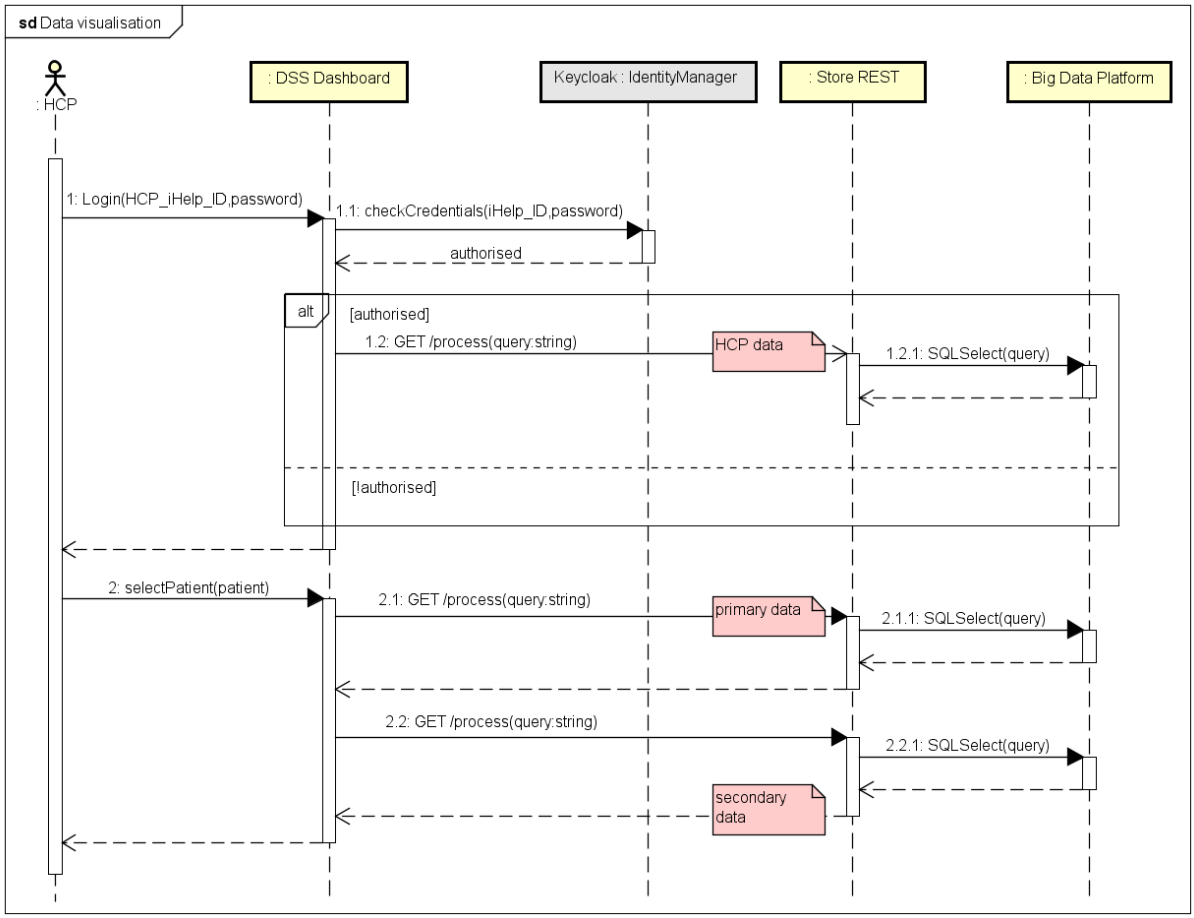

Figure 15: Data visualisation sequence diagram.

3.3.8 Risk assessment

The sequence diagram shown in Figure 16 illustrates the sequence of messages exchanged among the DSS Dashboard, the Analytic Workbench, the Personalised Predictor, and the Predictor & Risk Identifier to estimate the individual risk of developing Pancreatic Cancer, on request of the Health Care Professional.

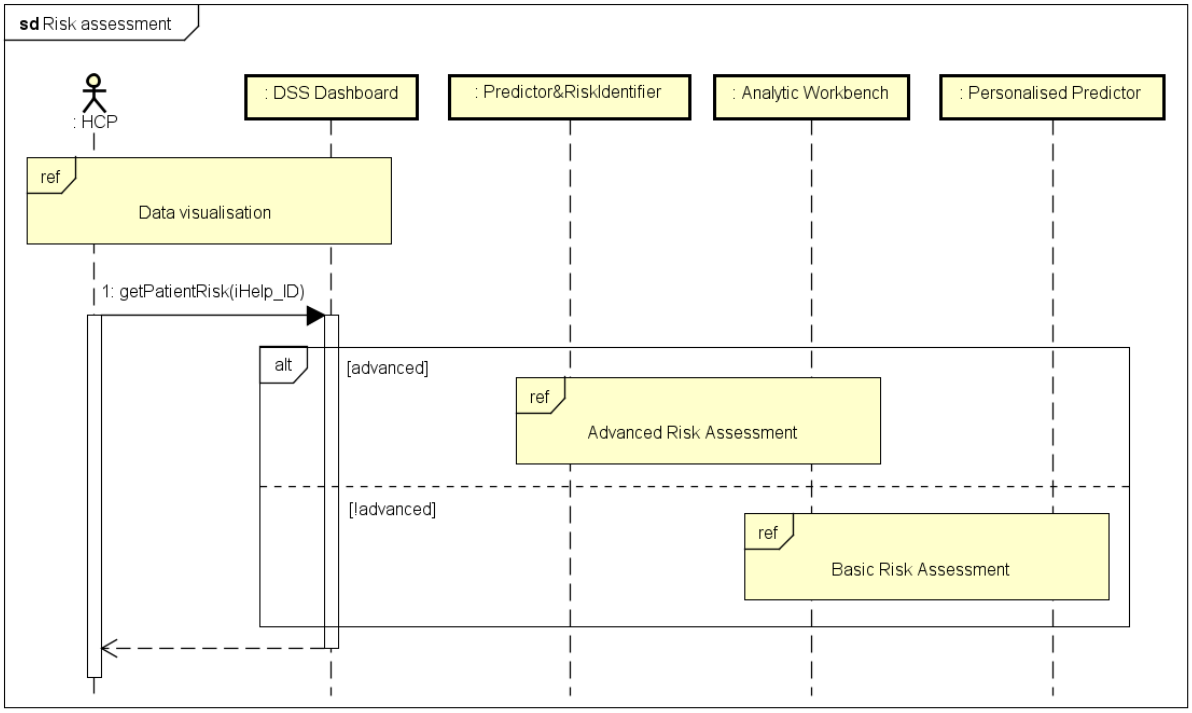

Figure 16: Risk assessment sequence diagram.

The exchange of messages is started by the Health Care professional that, after having logged in the DSS Dashboard and visualised patient's data (see *Data visualisation* sequence diagram in section 3.3.7), requests the calculation of the risk of contracting PC for the patient.

Two types of risk assessment are available in the iHelp platform and have to be chosen by the HCP:

- *Basic risk assessment* (see Figure 17), performed by the Personalised Predictor via the Analytic Workbench using models trained with historical (primary) data.
- *Advanced risk assessment* (see Figure 18), offered by the Predictor & Risk Identifier via the Analytic Workbench with models refining the above ones with secondary data.

In both cases, the risk assessment is performed automatically using the preferred model set according to the *Select a preferred risk prediction model* sequence diagram available in section 3.3.19. The HCP is involved in the risk assessment process only to provide missing data.

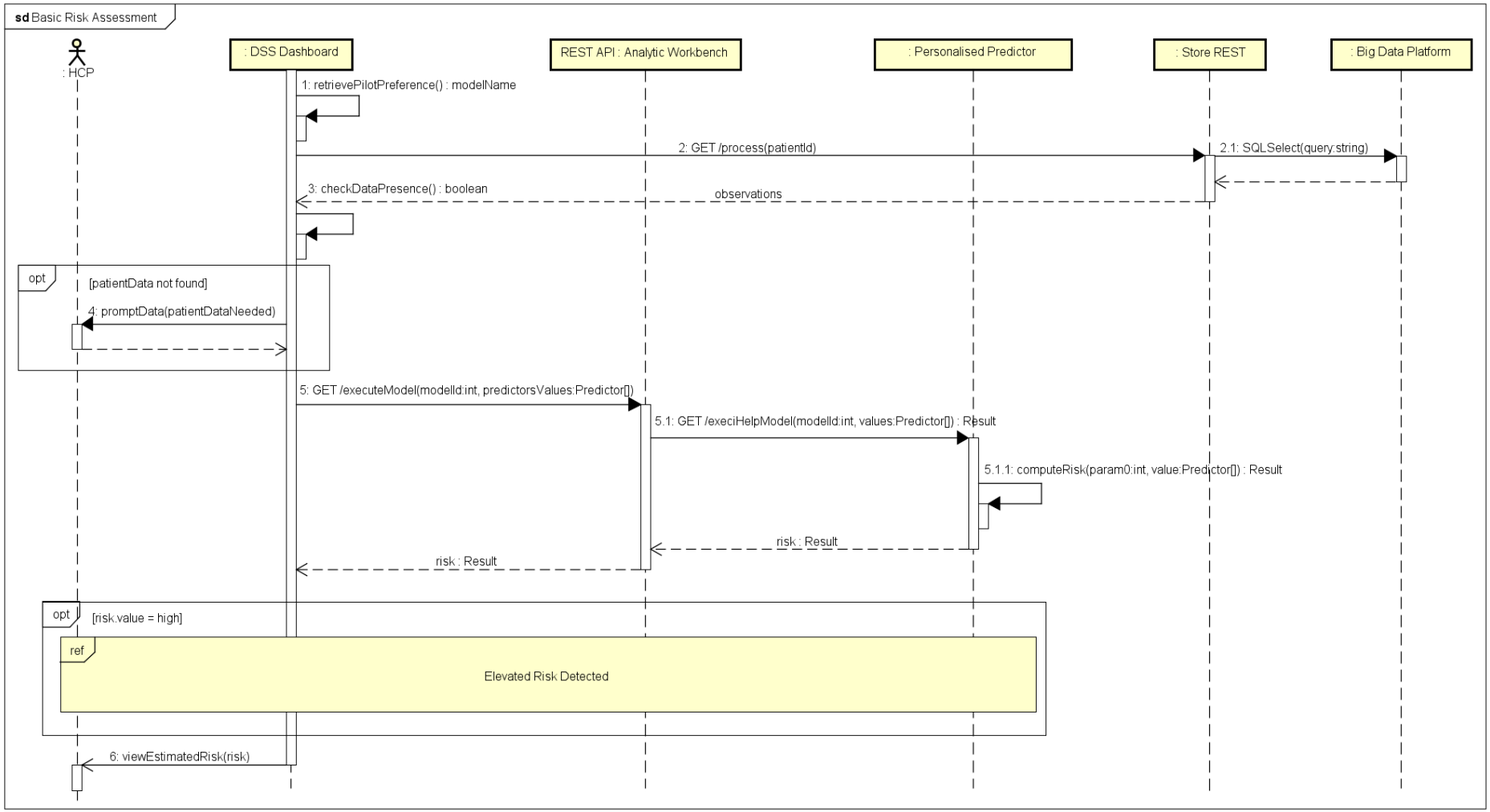

Figure 17: Basic risk assessment sequence diagram.

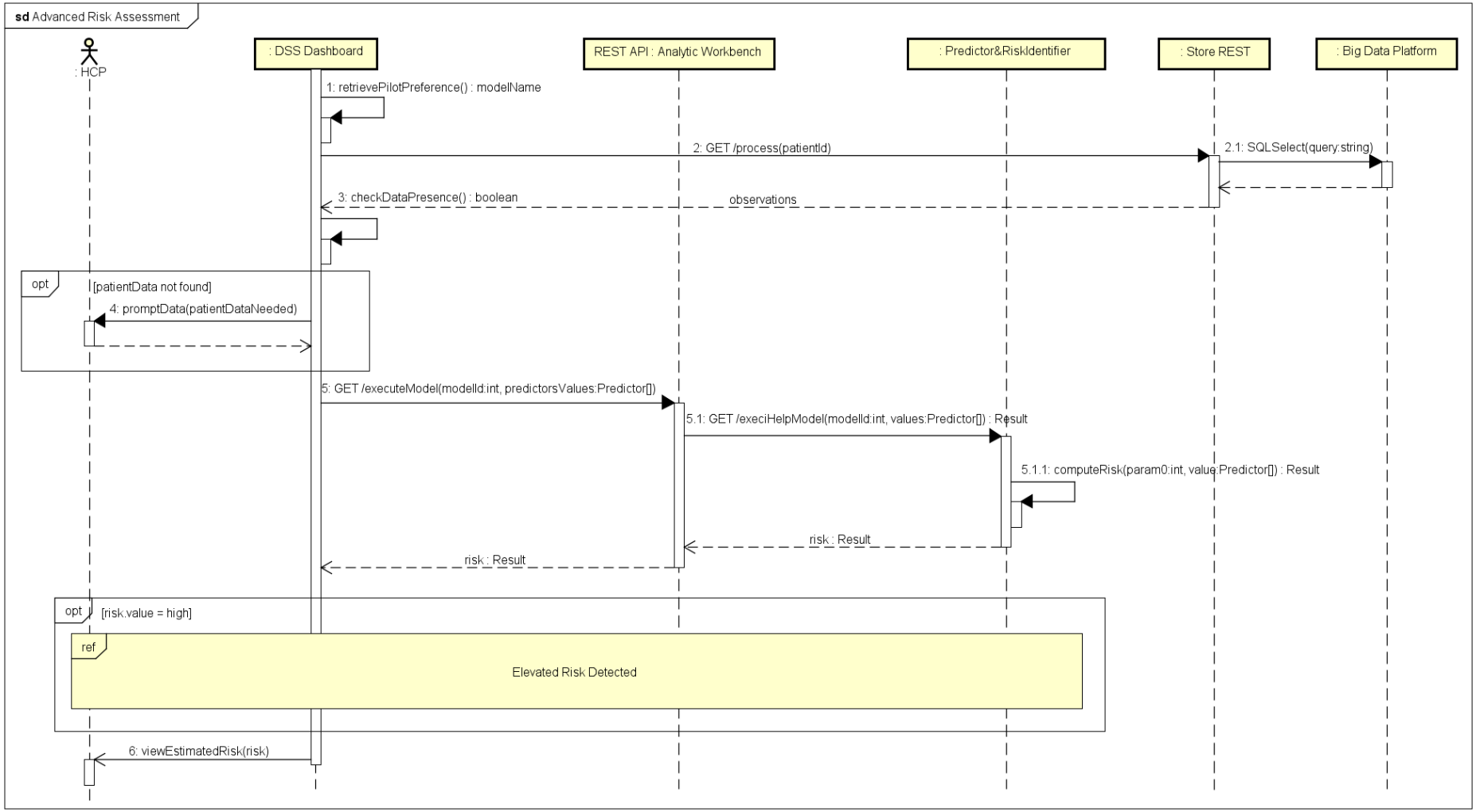

Figure 18: Advanced risk assessment sequence diagram.

The basic risk assessment, involving the DSS Dashboard, the REST API of the Analytic Workbench, the Personalised Predictor and the Big Data Platform, is initiated by the DSS Dashboard, that internally retrieves the risk prediction model preferred by a pilot (chosen in *Select a preferred risk prediction model* use case, whose sequence diagram is reported in section 3.3.19), that will be used to estimate the individual risk of contracting PC. The DSS Dashboard then retrieves patient data from the Big Data Platform via the Store REST and obtains the patient observations, including the name of the attributes that should have non null values, i.e. patient data required to perform risk assessment. The DSS Dashboard internally checks whether these data are available; if not the HCP is prompted to provide them. When the DSS Dashboard has all required data, it invokes the endpoint `GET /executeModel` made available by the Analytic Workbench REST API, by specifying the `modelId`, i.e. the identifier of the risk prediction model, and the `predictorValues`, i.e. the required patient data. The Analytic Workbench then calls the `GET /execiHelpModel/{modelId}/{patientData}` endpoint of the Personalised Predictor, that computes the risk and returns the result, representing the assessed risk level. If the risk level is high, the *Elevated Risk Detected* sequence of messages (see section 3.3.12) is triggered. At the end of the risk assessment process, the estimated risk level is visualised to the dashboard of the Health Care Professional.

The advanced risk assessment process, involving the DSS Dashboard, the REST API of the Analytic Workbench, the Predictor & Risk Identifier and the Big Data Platform, is started by the DSS Dashboard and is very similar to the one followed to carry out the basic risk assessment. The main differences are:

- The risk prediction models used to perform risk assessment are different.
- The required patient data are different.
- The component computing the risk, called by the Analytic Workbench REST API using the same endpoint, is the Predictor & Risk Identifier (instead of the Personalised Predictor involved in the basic risk assessment).

The Predictor & Risk Identifier returns the result containing the estimated risk level. If such a level is high, the *Elevated risk detected* process, illustrated in section 3.3.12, is triggered and the assessed risk is visualised in the dashboard of the HCP.

3.3.9 Communication of risk

The diagram depicted in Figure 19 models the sequence of calls done by iHelp components to communicate the level of risk of contracting PC to the individual participating in the pilot after the risk assessment (see section 3.3.8) has resulted in a high level of risk estimation (see section 3.3.12).

The involved iHelp components are the DSS Dashboard, The Personalised Advisor, the Mobile App, the Secondary Data Pre-processor, the Store REST and the Big Data Platform. The process is started by the DSS Dashboard, that invokes the `communicateRisk(risk,iHelp_ID)` method of the Personalised Advisor to convey the risk to the patient identified by `iHelp_ID` and the Personalised Advisor visualises the communication in the Mobile App of the patient. Then, the DSS Dashboard stores the message containing the communication of risk in the Big Data Platform via the Store REST component.

When the risk message is received by the patient, the Mobile App invokes the Secondary Data Pre-processor to set the message as “received” and the Secondary Data Pre-processor changes message status in the system by starting the secondary data ingestion pipeline, described in section 3.3.4.

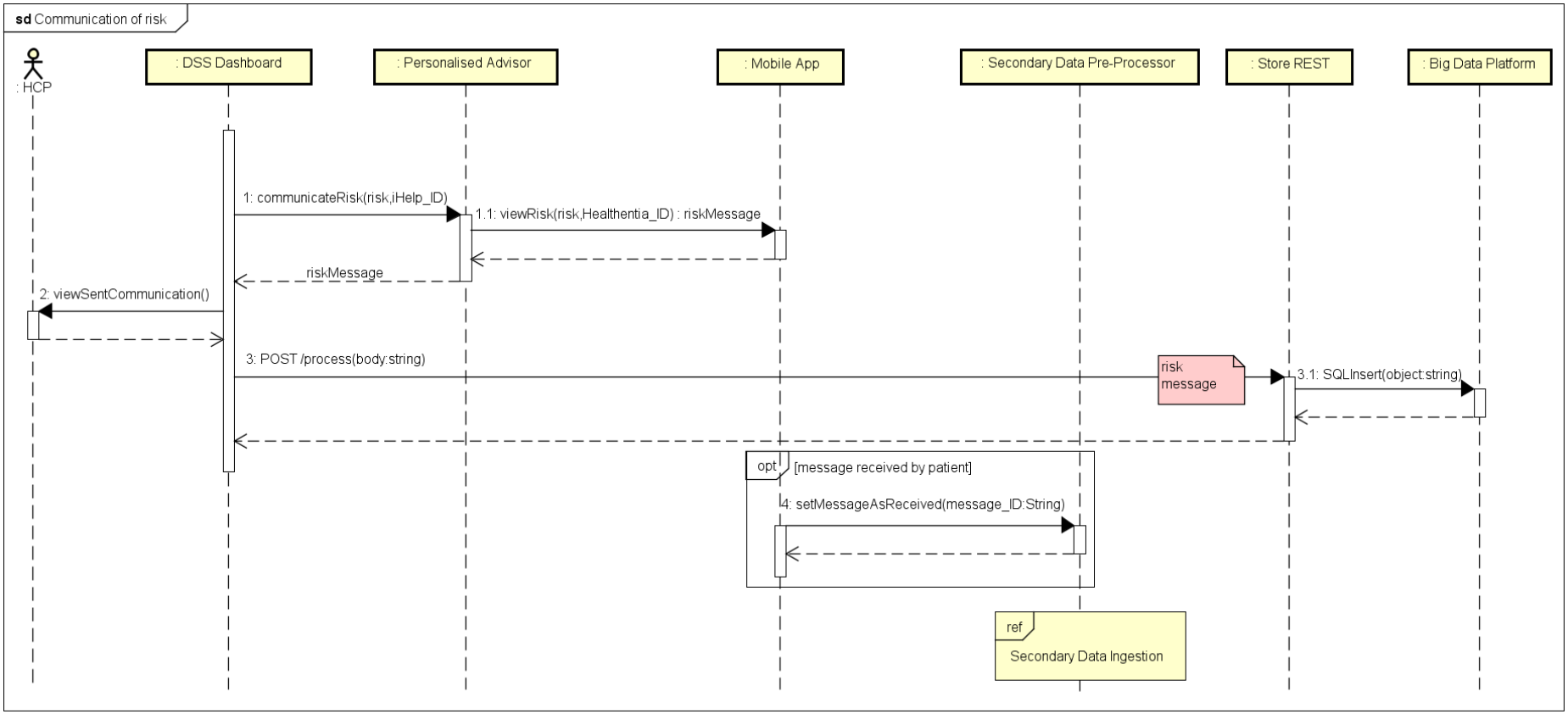

Figure 19: Communication of risk sequence diagram.

3.3.10 Request test and samples

The diagram reported in Figure 20 identifies the sequence of messages exchanged among iHelp components to allow the Health Care Professional to generate a list of tests that the patient has to perform. This event occurs after a high risk of contracting PC is estimated (see section 3.3.12), in order to have a more precise risk assessment.

The involved iHelp components are the DSS Dashboard, the Personalised Advisor, the Mobile App, the Secondary Data Pre-processor; the external laboratory in charge of executing the requested tests is also implicated.

The process is initiated by the DSS Dashboard that, after an elevated risk of developing PC is assessed, should trigger the delivery of the tests request to the patient. If the list of tests needs HCP approval before being sent to the interested individual, the clinician is prompted with a default catalogue of tests and samples requiring approval. Even if the clinician accepts them, he/she is anyway asked if any change is required: if this is the case, the DSS Dashboard allows the HCP to edit the proposed tests.

At the end of tests approval process, or in case this approval is not needed, the DSS Dashboard views the list of test that will be requested to the patient and invokes the Personalised Advisor to deliver the test request to the patient identified by `iHelp_ID` and the Personalised Advisor notifies the request to the patient in the Mobile App. When the request is received by the patient, the Mobile App calls the Secondary Data Pre-processor to set the request message as “received” and the Secondary Data Pre-processor enters it in the system by starting the secondary data ingestion pipeline, described in section 3.3.4.

Afterwards, the individual goes to a (external) Test Laboratory, that executes the required tests and samples and generates the results, that are inserted in the platform with the primary Ingestion pipeline, delineated in section 3.3.3.

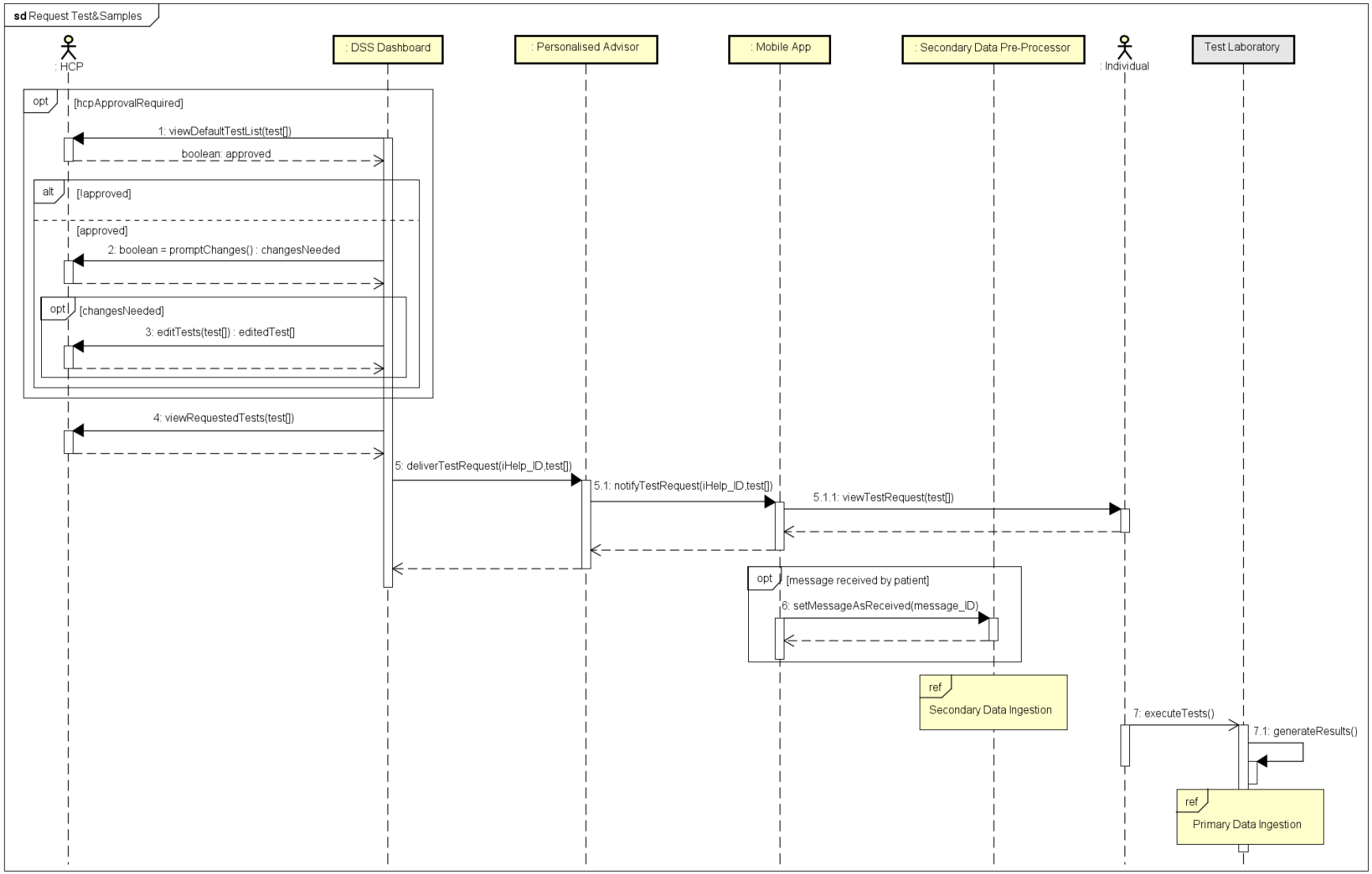

Figure 20: Request tests and samples sequence diagram.

3.3.11 Plan a visit or control

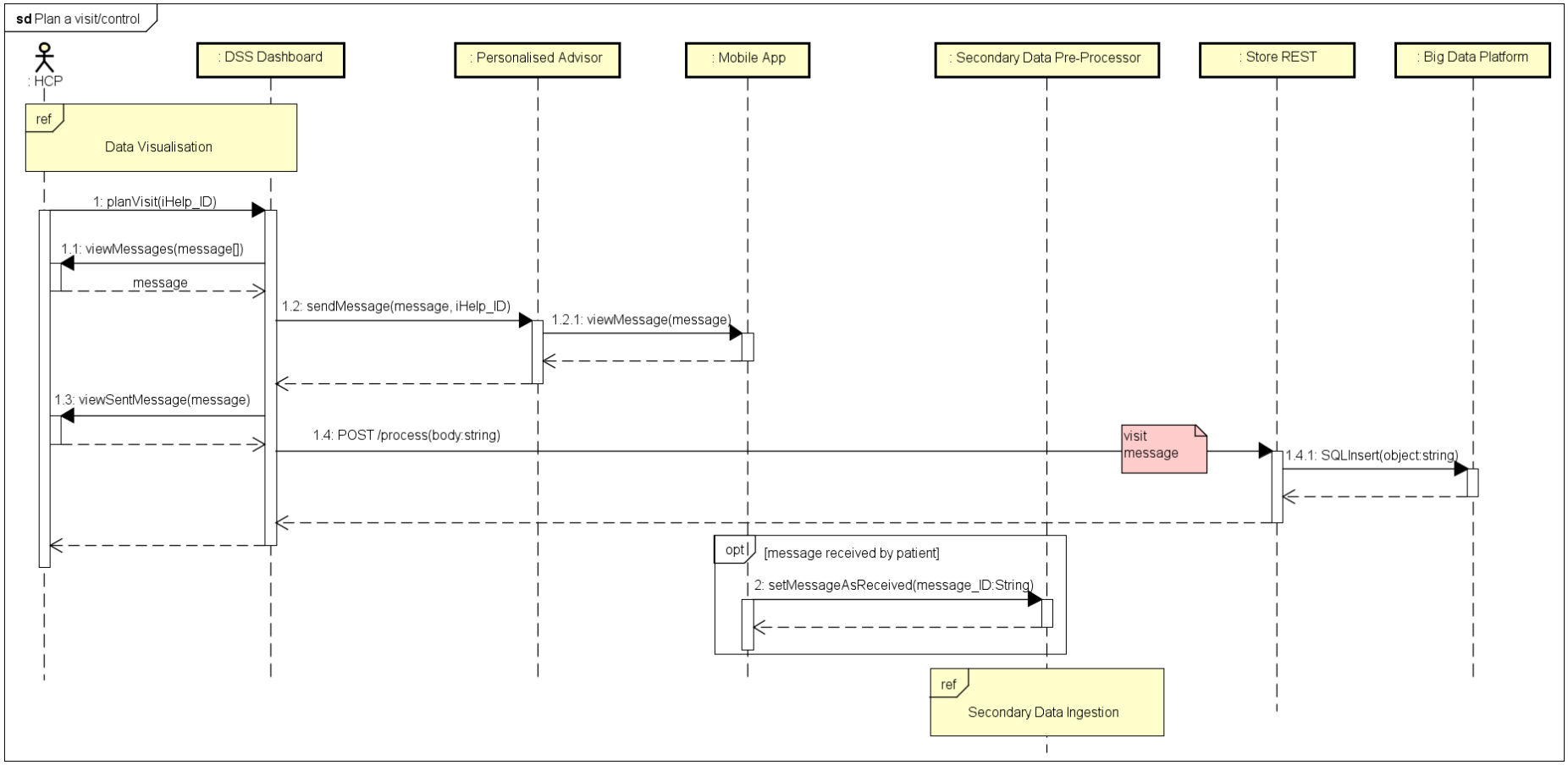

Figure 21: Plan a visit or control sequence diagram.

The sequence diagram displayed in Figure 21 identifies the messages exchanged among iHelp components to plan a visit or control, i.e. to send a message regarding a visit/control to the patient using the iHelp platform.

The involved components are the DSS Dashboard, the Personalised Advisor, the Mobile App, the Secondary Data Pre-processor, the Store REST and the Big Data Platform.

The *Plan a visit or control* process is initiated by the Health Care Professional that, after having logged in the DSS Dashboard and visualised patient's data (see *Data visualisation* sequence diagram in section 3.3.7), requests the planning of a visit and is prompted with a list of available messages that can be set to the patient.

The clinician selects a message, the DSS Dashboard invokes the Personalised Advisor to send it to the patient identified by `iHelp_ID` and the Personalised Advisor visualises the message in the Mobile App of the patient. The DSS Dashboard views the sent message to the patient then stores it in the Big Data Platform via the Store REST component.

When the message is received by the patient, the Mobile App invokes the Secondary Data Pre-processor to set the message as "received" and the Secondary Data Pre-processor changes message status by starting the secondary data ingestion pipeline, described in section 3.3.4.

3.3.12 Elevated risk detected

The sequence diagram shown in Figure 22 represents the messages exchanged among iHelp components when the estimation of the risk of developing Pancreatic Cancer (see section 3.3.8), required by the Health Care Professional via the DSS Dashboard, returns a high value.

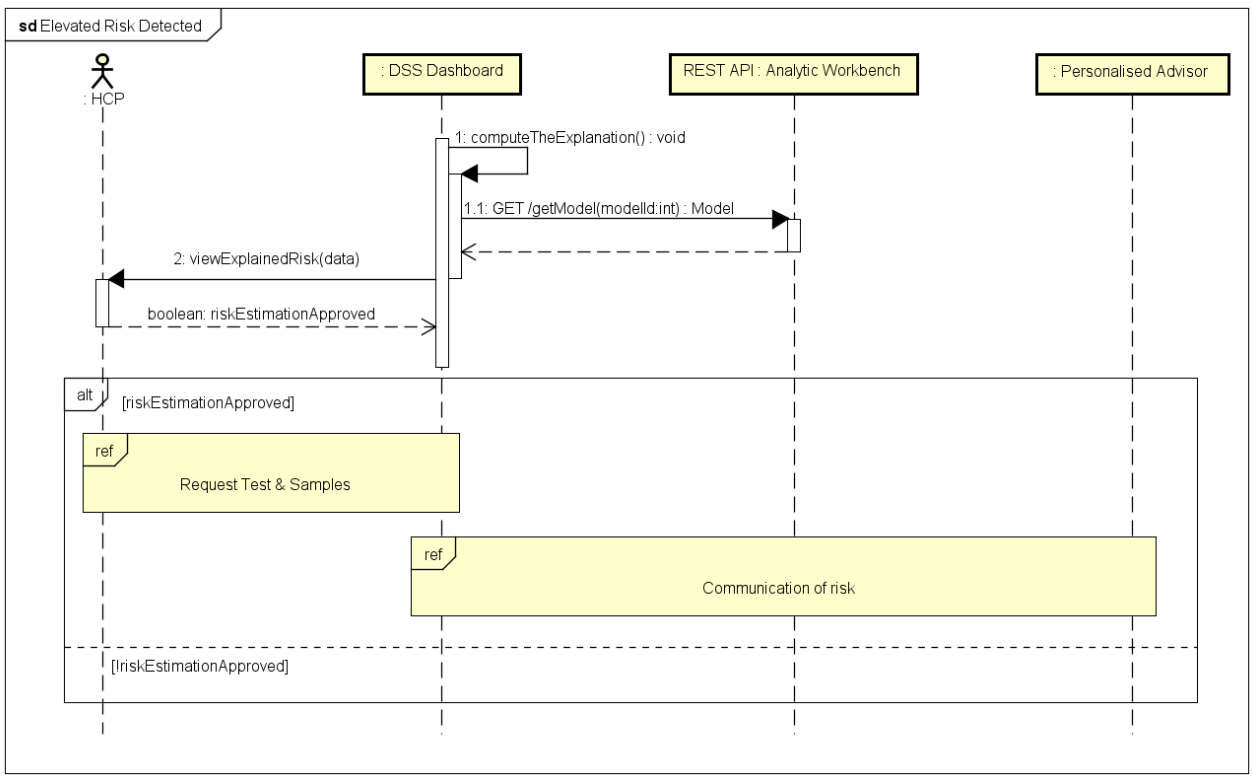

Figure 22: Elevated risk detected sequence diagram.

The involved components are the DSS Dashboard, the Analytic Workbench and the Personalised Advisor. The sequence is started by the DSS Dashboard that, in case a high-risk value is estimated, internally computes machine learning model explanation, invokes the `GET /getModel/{modelId}` endpoint of the Analytic Workbench to retrieve the model used for risk assessment and visualises the result of the risk assessment and the related explanation, asking the clinician to approve the estimation. If the risk estimation is approved by the Health Care Professional, further tests are requested (see *Request test & samples* sequence diagram in section 3.3.10) and then the assessed risk level is communicated to the patient (see *Communication of risk* sequence diagram in section 3.3.9).

3.3.13 Risk mitigation/treatment planning

The sequence diagram shown in Figure 23 details how iHelp components collaborate to allow the Health Care Professional to plan a risk mitigation advice or a treatment, i.e. a list of tasks or goals to achieve, for a patient estimated at high risk of contracting PC (or already diagnosed with PC and undergoing treatment). Once laboratory tests of a patient, requested by the Health Care Professional (see section 3.3.10) after a high risk evaluation (see section 3.3.12), are processed and primary data and questionnaires info are acquired, the risk mitigation planning is started.

The involved iHelp components are the DSS Dashboard, the Monitoring and Alerting and the Big Data Platform.

The sequence is initiated by the Health Care Professional, that accesses the DSS Dashboard to visualise the data of a patient (see section 3.3.7). The HCP then uses the Monitoring and Alerting to manage the mitigation plan, i.e., a set of alert rules associated to the patient. The alert rules (also called goals if is something standard that will be stored in the Big Data Platform) contain the conditions to be met when comparing monitored parameter target values/thresholds defined in the personalized advices against real values. If the conditions are met, an alert is triggered and a message is sent to the patient (see section 3.3.16).

The HCP first select a mitigation plan by invoking the Monitoring and Alerting. Then the clinician can select the rules suitable for the patient by activating them (and leaving not recommended rules deactivated). The HCP cannot creates new rules but only activate/deactivate the existing ones. For each rule to be activated the HCP can optionally:

- change the rule threshold by invoking the `changeRuleTreshold(ruleId)` method of the Monitoring and Alerting
- set that the alert message needs the HCP approval before being delivered to the patient with the `setHCPApprovalRequired(ruleId)` method of the Monitoring and Alerting

After having activated the desired rules, the Health Care Professional saves the mitigation plan and the Monitoring and Alerting stores the set of goals of the patient in the Big Data Platform.

To add new rules on HCP request, the Monitoring and Alerting calls the `insertRule(AlertRule)` method of the Big Data Platform.

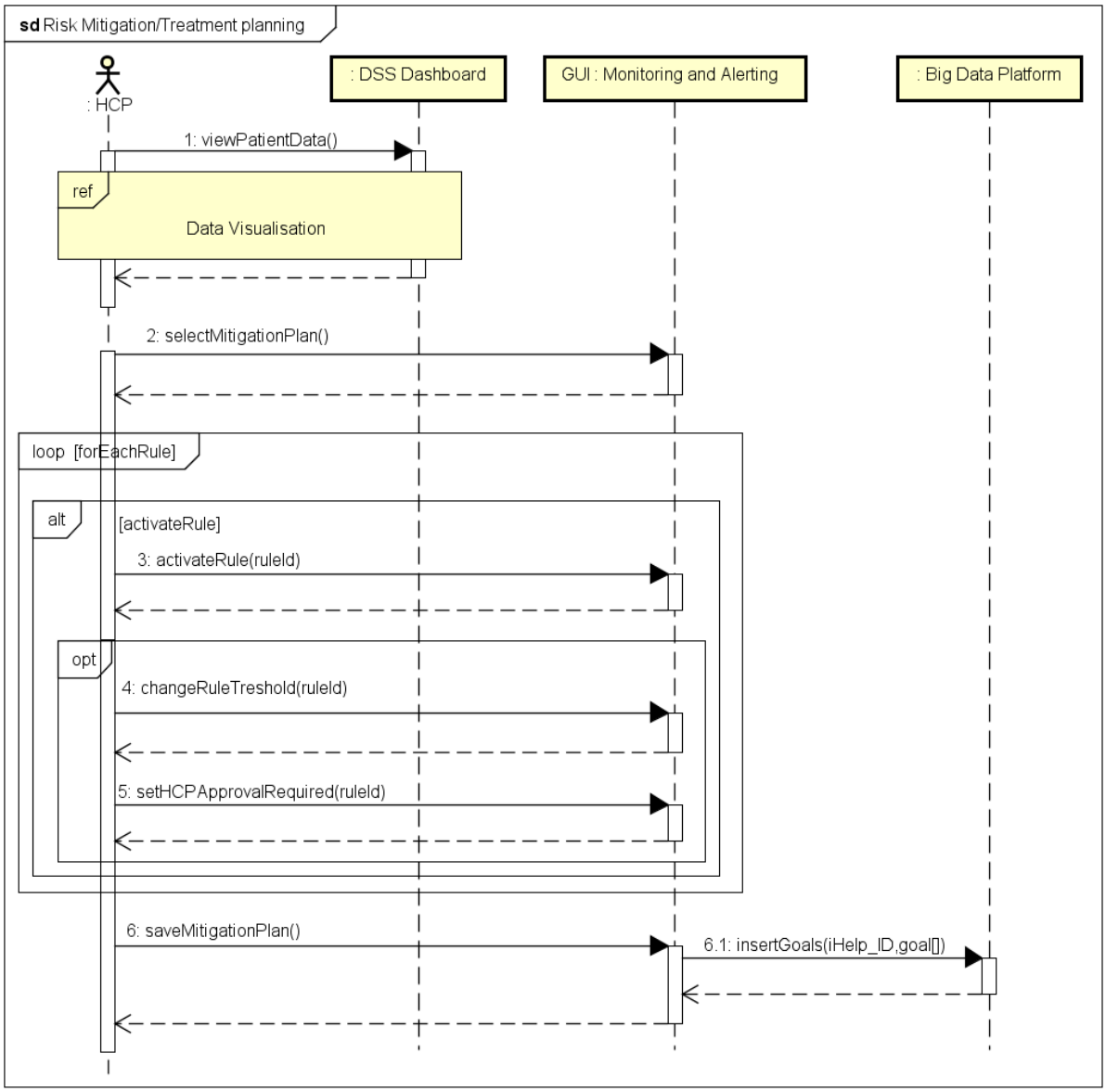

Figure 23: Risk mitigation/treatment planning sequence diagram.

3.3.14 Monitoring

The diagram in Figure 24 describes the sequence of calls occurring to monitor high-risk individuals in order to estimate if their behaviour is in line with the mitigation plan. Secondary data are continuously checked, aggregated and compared with personalized goals, i.e., target values defined in the personalised advices; if the goals are not achieved the alerting is activated. The alerting functionality is embedded in the *Risk Mitigation Delivery* sequence diagram (see section 3.3.16).

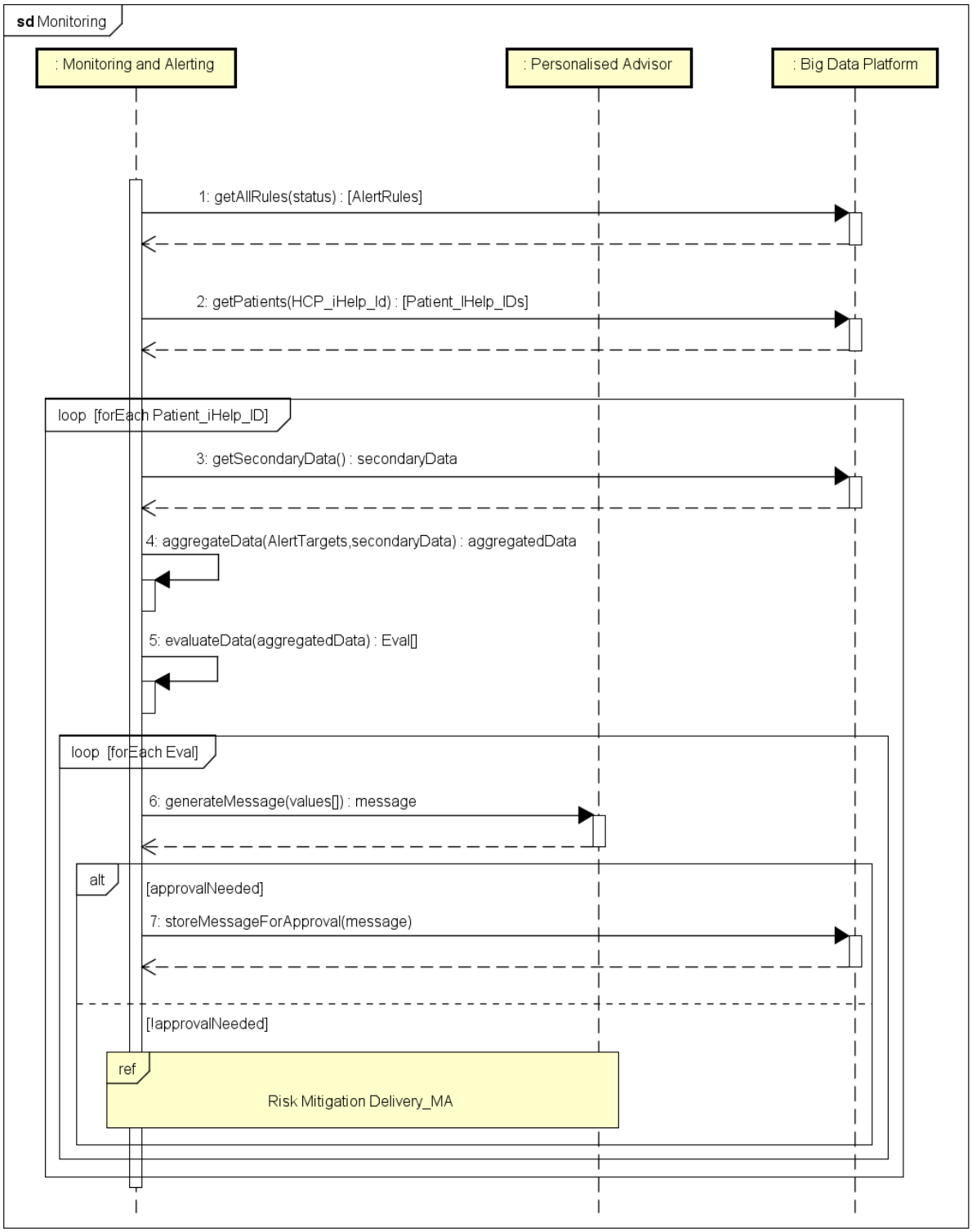

Figure 24: Monitoring sequence diagram.

The involved iHelp components are the Monitoring and Alerting, the Personalised Advisor and the Big Data Platform. The Monitoring and Alerting component invokes the Big Data Platform to retrieve all stored alert rules and all patients treated by a specific Health Care Professional (identified by the `HCP_iHelp_Id`).

Then for each patient (identified by `Patient_iHelp_ID`) , the Monitoring and Alerting:

1. Invokes the Big Data Platform to obtain raw secondary data collected by the patient
2. Internally aggregates row data if rule uses comparison of aggregated data against target
3. Internally evaluates aggregated data against target and produces evaluations. For each evaluation, the Monitoring and Alerting invokes the Personalised Advisor to generate alert messages that will be delivered to interested patients.

When managing rules in a mitigation plan, the HCP can set that an alert message needs their approval before being sent to the patient (see section 3.3.13), as pointed out in the requirements (see D2.3 (M., A., C., + 22), section 4.10.3). In case the HCP approval is required, the message returned by the Personalised Advisor to the Monitoring and Alerting is stored for future approval by calling the Big Data Platform. If no approval is needed, the message is finally delivered to the patient, as designed in the *Risk Mitigation Delivery* sequence diagram (see section 3.3.16).

3.3.15 Advice review

The diagram shown in Figure 25 represents the sequence of messages exchanged among the DSS Dashboard, the Personalised Advisor, the Store REST and the Big Data Platform to allow the Health Care Professional to endorse the suggestion proposed by the Monitoring and Alerting (see *Monitoring* sequence diagram in section 3.3.14) and enable the actual delivery of the messages to the interested patient(s).

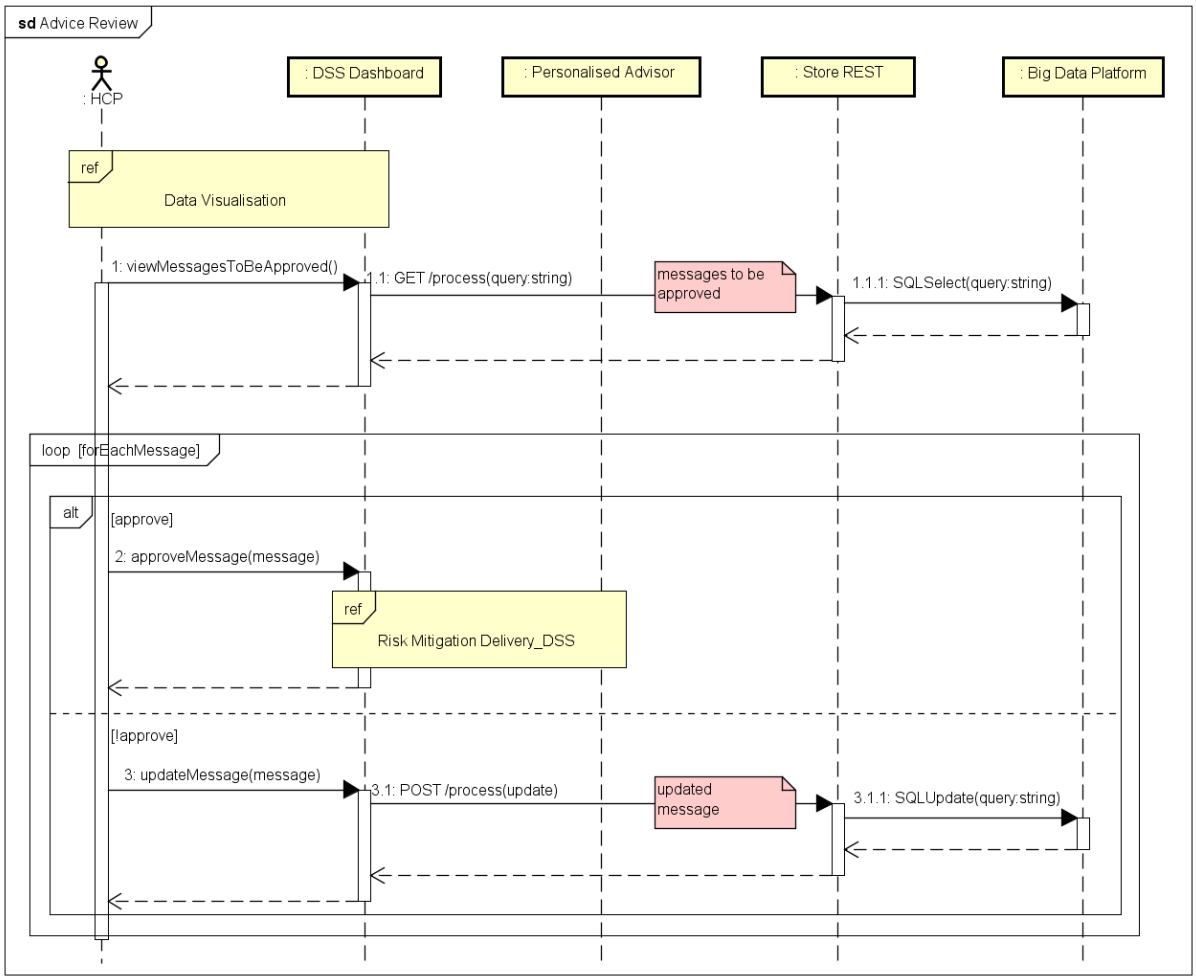

Figure 25: Advice review sequence diagram.

The Health Care Professional initiates the sequence by accessing the DSS Dashboard (see *Data visualisation* sequence diagram in section 3.3.7) that visualises stored messages needing HCP approval (as explained in section 3.3.14), retrieved by invoking the `GET /process(query)` endpoint made available by the Store REST, that then executes a SQL `Select` in the Big Data Platform.

For each message, the HCP decides whether to approve it or not. If the message is approved, the DSS Dashboard invokes the Personalise Advisor to deliver the message to the patient, as illustrated in the *Risk Mitigation Delivery* sequence diagram (see section 3.3.16). If the message is not approved by the HCP, in order to preserve it for future analytical purposes, the DSS Dashboard stores it with "rejected" status, by invoking the `POST /process(query)` endpoint made available by the Store REST, that then executes a SQL `Update` in the Big Data Platform.

3.3.16 Risk mitigation delivery

The diagrams shown in Figure 26 and Figure 27 identify the sequence of messages exchanged among iHelp components to deliver to a patient messages/advices for the mitigation plan. During the monitoring of a patient (see section 3.3.14), if the goals specified in the mitigation plan (whose setup is detailed in section 3.3.13) are not achieved, the alerting functionality is activated and the risk mitigation delivery process deals with the delivery of alert messages to the patient. When personalised alert messages are created, it is possible to set that they require the approval of the clinician before actually delivering them to the interested individual. In case the HCP approval is required, the alert message is stored until the Health Care Professional approves following the *Advice review* process (see section 3.3.15) and the alert is actually delivered to the interested patient.

The *Risk Mitigation Delivery_DSS* diagram is related to the risk mitigation delivery process triggered by the DSS Dashboard during the advice review. This process, involving the DSS Dashboard, the Personalised Advisor, the Store REST, the Mobile App, the Secondary Data Pre-processor and the Big Data Platform, is initiated by the DSS Dashboard, that invokes the Personalised Advisor to send a message approved by the HCP to the patient. The Personalised Advisor displays the message in the Mobile App of the patient, then the DSS Dashboard updates the message status to "approved" in the Big Data Platform via the Store REST component. When the message is received by the patient, the Mobile App calls the Secondary Data Pre-processor to set the message as "received" and the Secondary Data Pre-processor changes the message status by starting the secondary data ingestion pipeline, described in section 3.3.4.

The *Risk Mitigation Delivery_MA* sequence diagram illustrates the risk mitigation delivered process triggered by the Monitoring and Alerting during the monitoring of a patient, in those cases when alert messages do not need HCP approval and can be directly delivered to the patient. It implicates the Monitoring and Alerting, the Personalised Advisor, the Mobile App, the Secondary Data Pre-processor and the Big Data Platform,

The sequence is started by the Monitoring and Alerting, that invokes the Personalised Advisor to send a message to the patient. The Personalised Advisor views the message in the Mobile App of the patient, then the Monitoring and Alerting calls the Big Data Platform to save it. When the message is received by the patient, the Mobile App calls the Secondary Data Pre-processor to set the message as "received" and the Secondary Data Pre-processor changes message status by starting the secondary data ingestion pipeline, described in section 3.3.4.

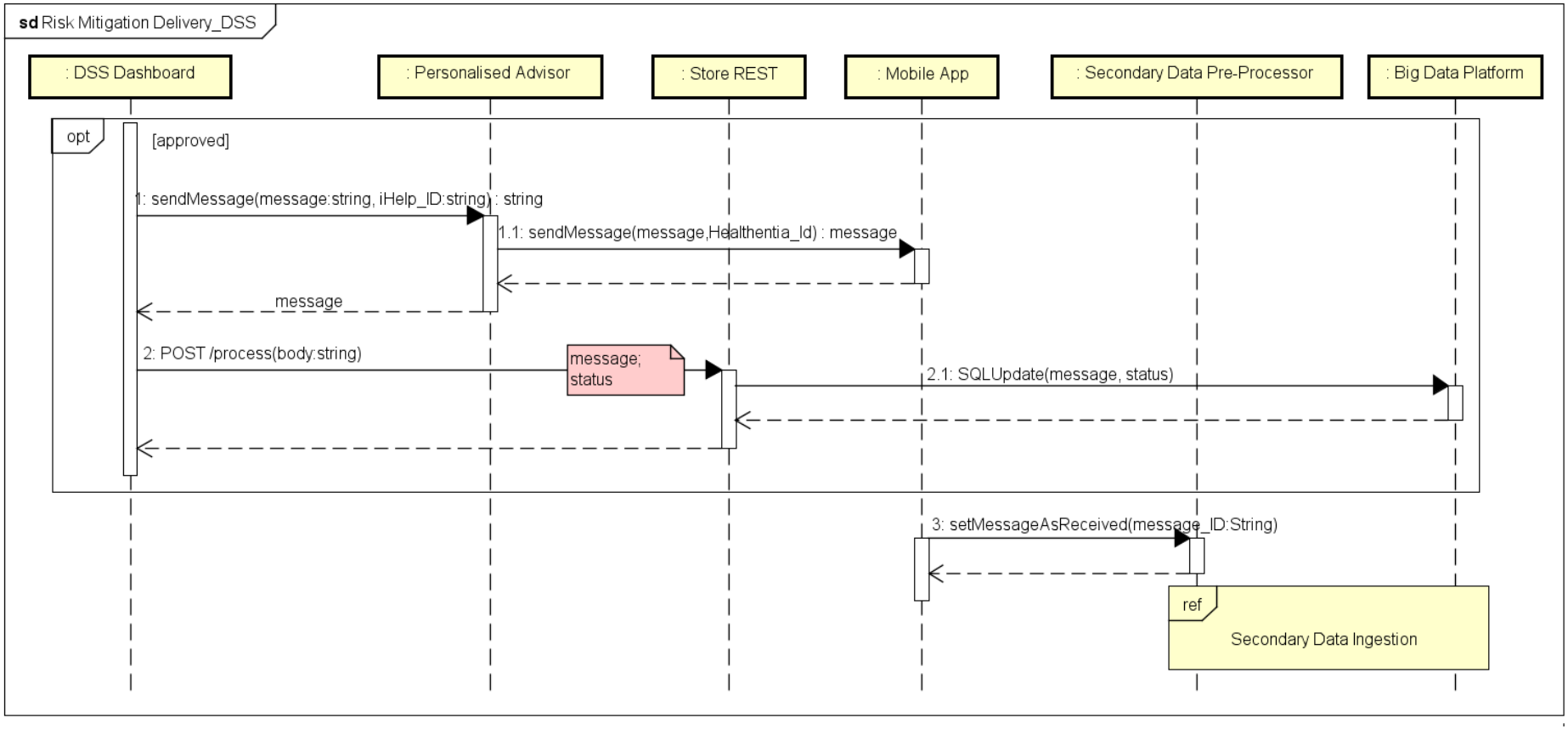

Figure 26: Risk mitigation delivery_DSS sequence diagram, illustrating the risk mitigation delivery process triggered by the DSS Dashboard

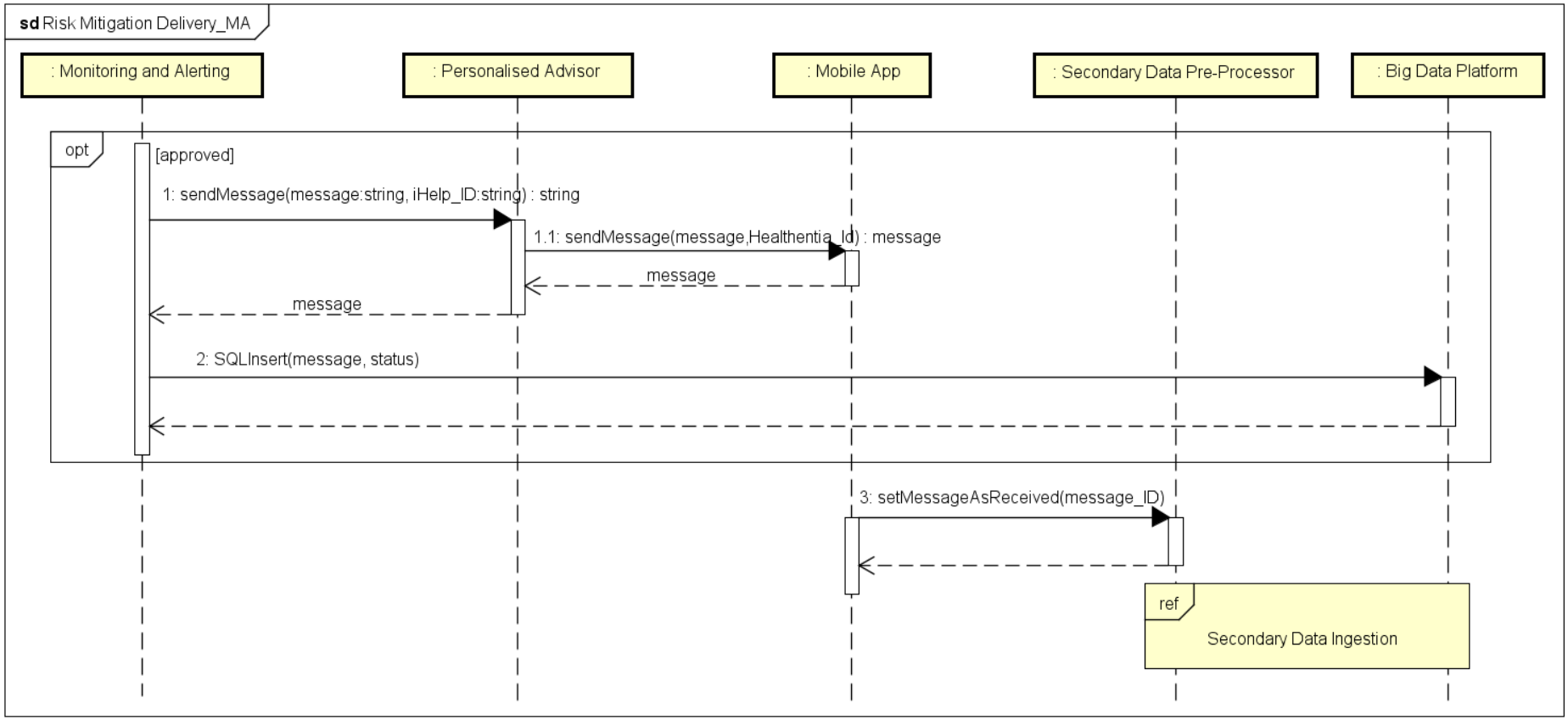

Figure 27: Risk mitigation delivery_MA sequence diagram, illustrating the risk mitigation delivery process triggered by the Monitoring and Alerting

3.3.17 Advice follow-up

The sequence diagram in Figure 28 illustrates the sequence of messages exchanged among iHelp components during the follow-up of the risk mitigation advice directed to high-risk of developing PC subjects. Using the DSS Dashboard, the clinician can analyse the effect of the sent messages on the patient's behaviour to estimate and evaluate which is the best pattern of intervention for the specific patient.

The involved iHelp components are the DSS Dashboard, the Monitoring and Alerting and the Big Data Platform.

The Health Care Professional initiates the sequence by accessing the DSS Dashboard (see *Data visualisation* sequence diagram in section 3.3.7). To allow the clinician to look at the progress of the mitigation plans, the DSS Dashboard offers one of the following options:

- *View patient statistics* allows to retrieve the statistics about a specific patient in a given period in terms of available primary and secondary data available and sent advice messages. To do that, the DSS Dashboard calls the `GET /impact-evaluator/getPatientStats(patientId,startDate,endDate)` endpoint of the Monitoring and Alerting that, in turn, retrieves required primary data (`getPrimaryData(patientId,startDate,endDate)`), secondary data (`getSecondaryData(patientId,startDate,endDate)`) and sent messages (`getSentMessages(patientId,startDate,endDate)`) from the Big Data Platform.
- *View all messages statistics*, allowing to visualise the statistics on all messages stored in a specific period, is made available in the DSS Dashboard by invoking the `GET /impact-evaluator GET /getMessages(startDate,endDate)` endpoint of the Monitoring and Alerting, that obtains all messages stored in the Big Data Platform using the `getAllMessages(startDate,endDate)` method.
- *View message versus patient category report*, including the statistics about the sent messages and the recipients in a given period. To do that, the DSS Dashboard invokes `GET /impact-evaluator/getReport(startDate,endDate)` the endpoint of the Monitoring and Alerting, that successively gets all sent messages (`getAllMessages(startDate,endDate)` method) and patients' data (`getPatients(patientId[])` method) from the Big Data Platform.

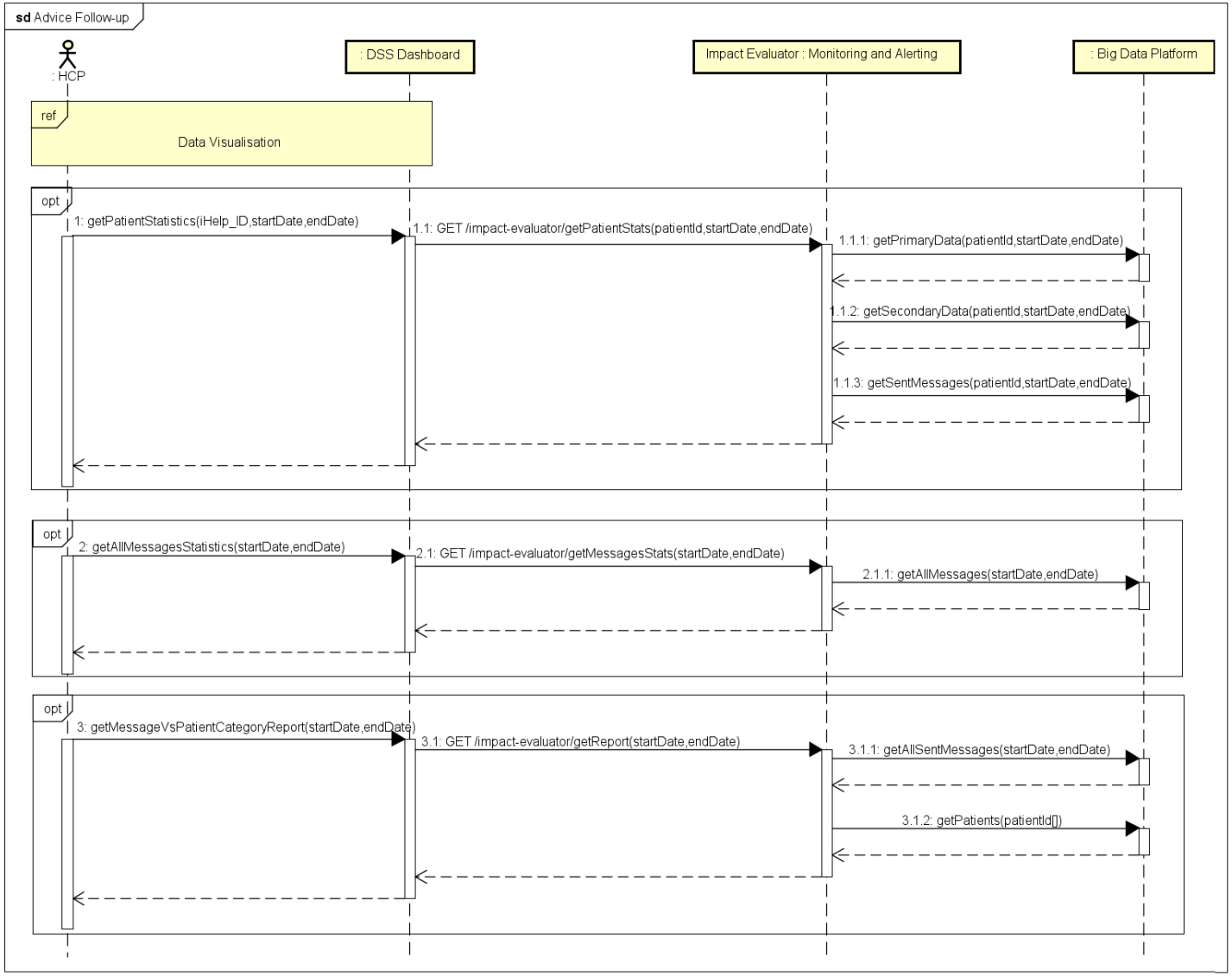

Figure 28: Advice follow-up sequence diagram.

3.3.18 Develop risk prediction model

Figure 29 shows the diagram illustrating the sequence of messages exchanged among iHelp components to allow the Model Builder to develop and train a risk prediction model with genomics and epidemiological data. The risk prediction model is used to assess the individual risk of contracting Pancreatic Cancer (see section 3.3.8).

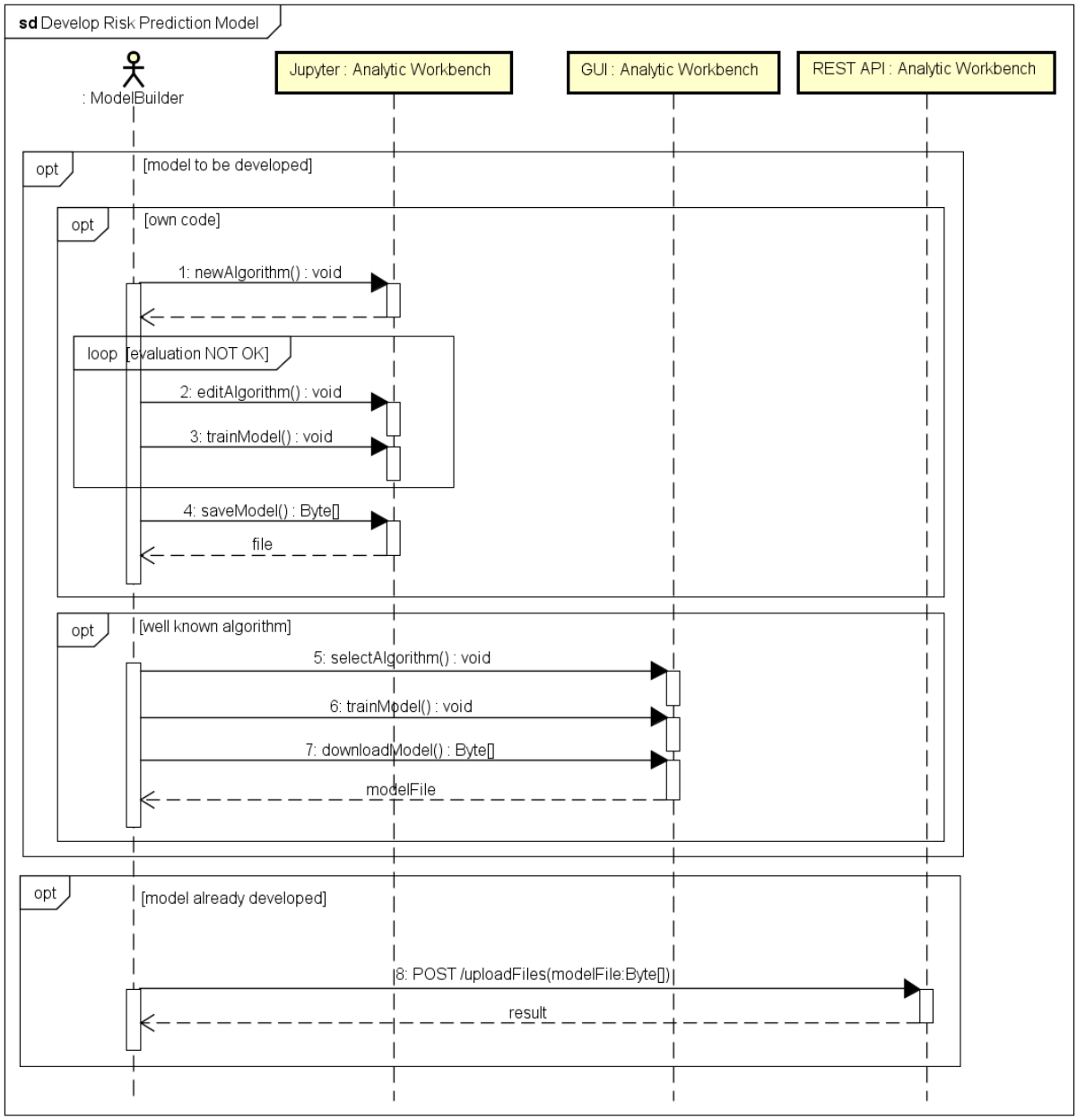

Figure 29: Develop risk prediction model sequence diagram.

The involved iHelp components are the GUI, Jupyter Notebook[7], and REST API modules of the Analytic Workbench.

[7] https://jupyter.org/

The process is started by the Model Builder, that has three different options to develop a risk prediction model:

1. Develop a new model using their own code. To do that the Model Builder uses the Jupyter Notebook to write the algorithm, that is then evaluated and trained with Jupyter until the evaluation is successful. Once the evaluation is ok, the Model Builder calls Jupyter to save the model in the Analytic Workbench and obtains the model file.
2. Develop a new model using a well know algorithm, such as scikit-learn, TensorFlow, H2o. To this purpose, the Model Builder accesses the Analytic Workbench GUI to select the algorithm to be used for training the model and train and test the model, that is then deployed in the Analytic Workbench. At the end, the Model Builder uses the GUI to download the file of trained model.
3. Upload an already developed and trained model by invoking the POST `/uploadFiles(modelFile)` endpoint of the REST API made available by the Analytic Workbench.

Model training, which here is modelled at a high level, can use data stored in the iHelp platform or external data as well.

A risk prediction model uploaded in the Analytic Workbench will be accessible by any iHelp component wanting to use it.

3.3.19 Select a preferred risk prediction model

The sequence diagram shown in Figure 30 describes the messages exchanged among iHelp components to allow the Model Builder, in agreement with the HCP, to select the preferred risk prediction model per pilot that will be used to perform risk assessment (see section 3.3.8). Risk estimation is requested by the Health Professional in the DSS Dashboard, that then automatically invokes the Analytic Workbench without involving the clinician, that does not have the necessary knowledge to choose the risk prediction model, required to compute the risk estimation. Therefore, there is the need to let the DSS Dashboard know which model will be used by default when calling the Analytic Workbench to start the risk assessment.

The involved components are the DSS Dashboard and the Analytic Workbench. The sequence is initiated by the Model Builder, that accesses the DSS Dashboard to retrieve all risk prediction models. The DSS Dashboard invokes the `GET /listModels` endpoint of the Analytic Workbench to obtain a list of available models.

The Model Builder views details of each model in the DSS Dashboard, that receives this information by calling the `GET /getModel(modelId)` endpoint of the Analytic Workbench, until a decision on the best prediction model is taken. The information displayed in the DSS Dashboard for each model is related to the patient observations, including the name of the attributes that should necessarily be provided to perform risk assessment. In order to make a decision, the Model Builder can discuss with the HCP; this consultation occurs offline and is out of the scope of this specification.

Once the Model Builder has chosen the preferred model of a pilot, the DSS Dashboard internally stores this setting, that will be used when the risk assessment is requested by the Health Care Professional. Different pilots can use different risk prediction models as preferred.

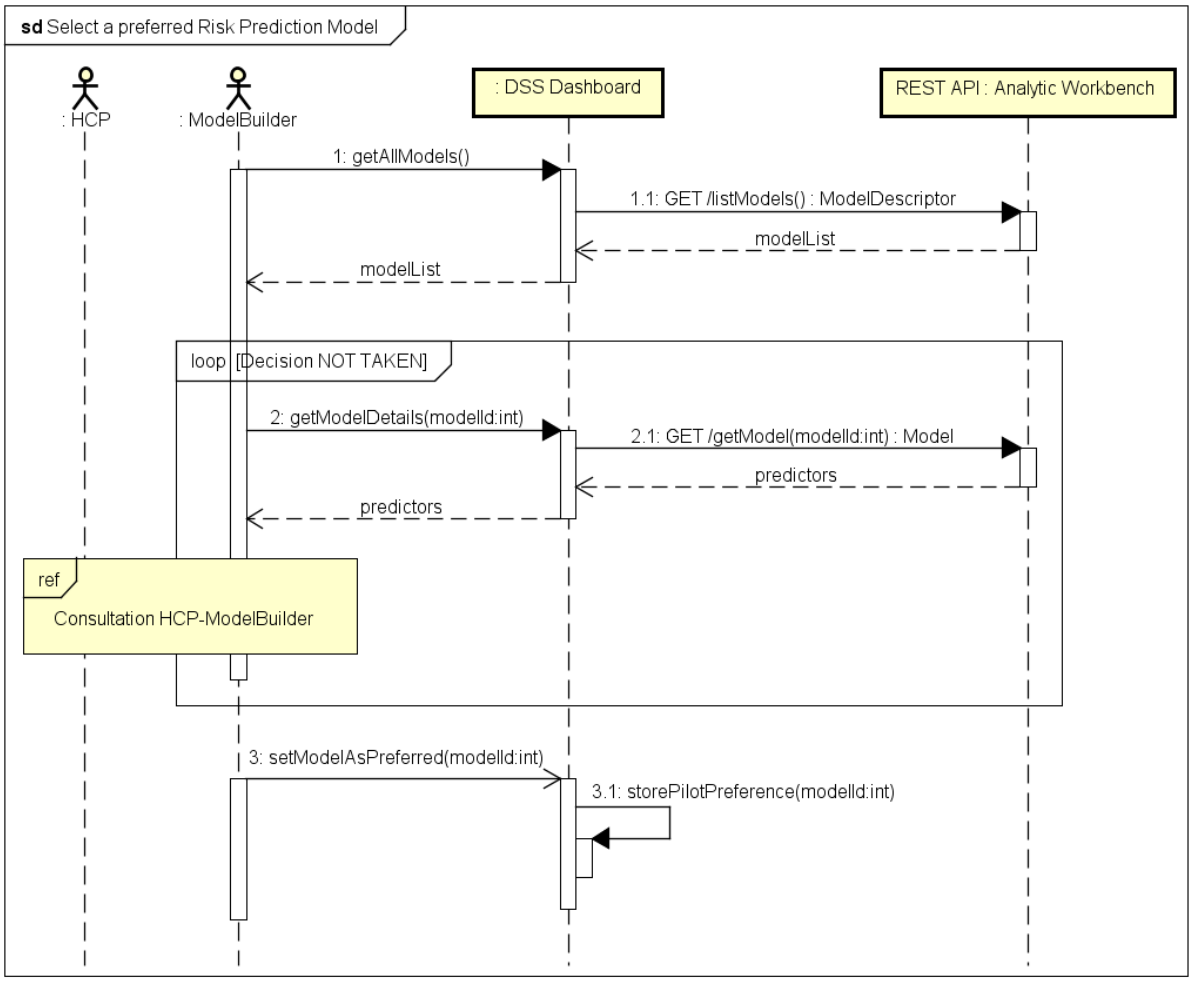

Figure 30: Select a preferred risk prediction model sequence diagram.

3.3.20 Social network data extraction

The sequence diagram depicted in Figure 31 illustrates the messages exchanged among iHelp components to allow the policy maker to understand the lifestyle trends and patterns on specific topics, in order to provide an impression of public sentiment which may help in developing new policies.

The involved components are the internal Social Analyser and the external Social Network API provided by Twitter and Reddit. The process is triggered by the Policy Maker, that initiates the Social Analyser and configures it by setting the:

- Involved social media
- Languages
- Keywords
- Rules

then starts the Social Analyser that first internally elaborates data, then continuously obtains data, by querying the Twitter API and the Reddit API, and create alerts.

The Policy Maker can:

- Optionally edit the keywords initially entered.
- View the alerts generated by the Social Analyser.

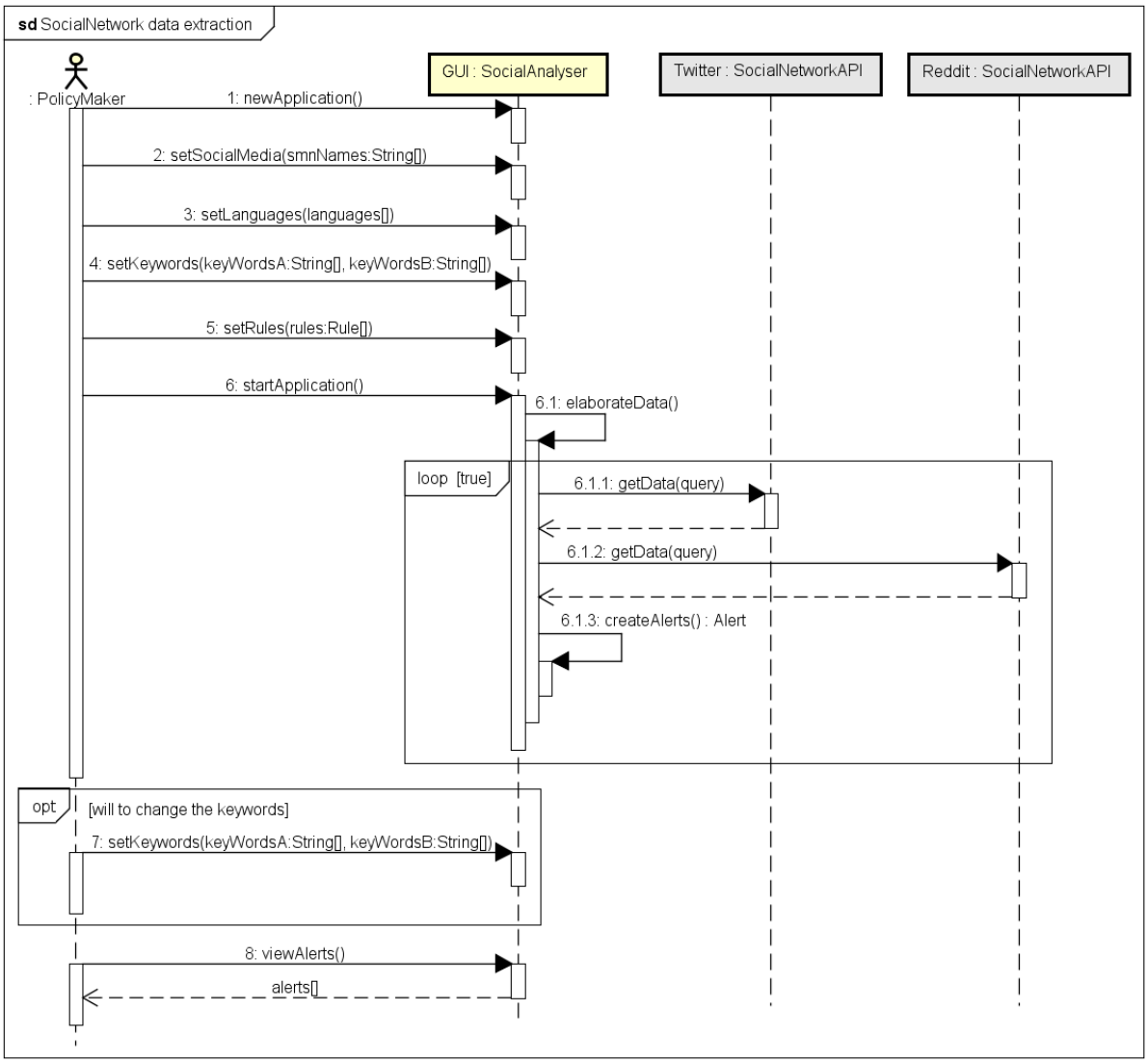

Figure 31: Social network data extraction sequence diagram.

3.4 Traceability matrix

The following figure and tables show the involvement of the iHelp components in the different use cases. This section details better which endpoint of each component is impacted by the use case in order to satisfy the behaviour modelled in the activity diagrams.

				Develop Risk Prediction Model	Risk Assessment	Request Tests and Samples	Elevated Risk Detected	Ingest Tests and Samples results	Risk Mitigation/ Treatment planning	Advice Review	Risk Mitigation Delivery	Monitoring	Advice Follow-up	Caption of profiling characteristics	Plan a visit or control	Data Visualisation	Communication of risk	Social Network data extraction	Patient enrolment using DSS Dashboard	Patient enrolment using Healthentia	Select a preferred risk prediction model
Data Capture Gateway	/datacapture	POST	/			X		X													
Data Gateway	/gateway	POST	/publish			X		X			X			X	X		X			X	
Data Gateway		Produce	T1			X		X			X			X	X		X			X	
Data Cleaner		Consume	T1			X		X			X			X	X		X			X	
Data Cleaner		Produce	T2			X		X			X			X	X		X			X	
Data Qualifier		Consume	T2			X		X			X			X	X		X			X	
Data Qualifier		Produce	T3			X		X			X			X	X		X			X	
Data Harmoniser		Consume	T3			X		X			X			X	X		X			X	
Data Harmoniser		Produce	T4			X		X			X			X	X		X			X	
HHR Importer		Consume	T4			X		X			X			X	X		X			X	
DSS Dashboard	GUI				X	X	X			X			X		X	X			X		X
Store REST	/	GET	/process		X					X			X			X			X		
Store REST	/	POST	/process							X	X				X				X		
Big Data Platform			CREATE			X		X	X		X			X	X		X		X	X	
Big Data Platform			READ		X				X	X		X	X			X			X		
Big Data Platform			UPDATE						X	X									X		
Big Data Platform			DELETE						X												
Secondary Data Pre-Processor	/v2/	POST	/account/login			X					X			X	X		X			X	
Secondary Data Pre-Processor	/v2/	POST	/account/RefreshToken			X					X			X	X		X			X	
Secondary Data Pre-Processor	/v2/	GET	/data-provider/answers			X					X			X	X		X			X	
Secondary Data Pre-Processor	/v2/	GET	/data-provider/physiological			X					X			X	X		X			X	
Secondary Data Pre-Processor	/v2/	GET	/data-provider/excercises			X					X			X	X		X			X	
Secondary Data Pre-Processor			setMessageAsReceived			X					X				X						
Personalised Advisor			deliverTestRequest(iHelp_ID,test[])			X															
Personalised Advisor			sendMessage(message)								X				X						
Personalised Advisor			generateMessage(values[])									X									
Analytic Workbench Jupyter				X																	
Analytic Workbench GUI				X																	
Analytic Workbench	/	GET	/listModels																		X
Analytic Workbench	/	GET	/getModel/{modelId}				X														X
Analytic Workbench	/	GET	/getAlgorithm/{modelId}	X																	
Analytic Workbench	/	GET	/executeModel/{modelId}/{patientData}		X																
Analytic Workbench	/	POST	/uploadFiles	X																	
Social Analyser			elaborateData															X			
Social Analyser			createAlerts															X			
Social Analyser			newApplication															X			
Social Analyser			setSocialMedia															X			
Social Analyser			setLanguages															X			
Social Analyser			setKeywords															X			
Social Analyser			setRules															X			
Social Analyser			startApplication															X			
Predictor	/	GET	/execiHelpModel/{modelId}/{patientData}		X																
Mobile App	GUI					X					X			X	X		X			X	
Monitoring and Alerting	/	POST	/ihelp-alert-data-aggregator/aggregate									X									
Monitoring and Alerting	/	POST	/ihelp-alert-evaluator/evaluate/monitoring									X									
Monitoring and Alerting	/	PUT	/ihelp-alert-evaluator/alerttarget/{id}						X												
Monitoring and Alerting	/	DELETE	/ihelp-alert-evaluator/alerttarget/{id}						X												
Monitoring and Alerting	/	GET	/ihelp-alert-evaluator/alertrule						X			X									
Monitoring and Alerting	/	POST	/ihelp-alert-evaluator/alertrule						X												
Monitoring and Alerting	/	GET	/impact-evaluator/getPatientStats										X								
Monitoring and Alerting	/	GET	/impact-evaluator/getMessagesStats										X								
Monitoring and Alerting	/	GET	/impact-evaluator/getReport										X								

Figure 32: Use Case/Methods Matrix.

Since the whole traceability matrix could be difficult to read, we have reported in the following a table per Use Case listing the involved endpoints.

Table 2: Components' methods involved in the *Develop Risk Prediction Model* Use Case

Develop Risk Prediction Model*			
Component	*Base URL*	*Type*	*Signature or Topic*
Analytic Workbench Jupyter			CREATE
Analytic Workbench GUI			READ
Analytic Workbench	/	GET	/getAlgorithm/{modelId}
Analytic Workbench	/	POST	/uploadFiles

Table 3: Components' methods involved in the *Risk Assessment* Use Case

Risk Assessment			
Component	*Base URL*	*Type*	*Signature or Topic*
DSS Dashboard	GUI		
Store REST	/	GET	/process
Big Data Platform			READ
Analytic Workbench	/	GET	/executeModel/{modelId}/{patientData}
Predictor	/	GET	/execiHelpModel/{modelId}/{patientData}

Table 4: Components' methods involved in the *Request Tests and Samples* Use Case

Request Tests and Samples			
Component	*Base URL*	*Type*	*Signature or Topic*
Data Capture Gateway	/datacapture	POST	/
Data Gateway	/gateway	POST	/publish
Data Gateway		Produce	T1
Data Cleaner		Consume	T1
Data Cleaner		Produce	T2
Data Qualifier		Consume	T2
Data Qualifier		Produce	T3
Data Harmoniser		Consume	T3
Data Harmoniser		Produce	T4
HHR Importer		Consume	T4
DSS Dashboard	GUI		
Big Data Platform			CREATE
Secondary Data Pre-Processor	/v2/	POST	/account/login
Secondary Data Pre-	/v2/	POST	/account/RefreshToken

Request Tests and Samples			
Processor			
Secondary Data Pre-Processor	/v2/	GET	/data-provider/answers
Secondary Data Pre-Processor	/v2/	GET	/data-provider/physiological
Secondary Data Pre-Processor	/v2/	GET	/data-provider/exercises
Secondary Data Pre-Processor			setMessageAsReceived
Personalised Advisor			deliverTestRequest(iHelp_ID,test[])
Mobile App	GUI		

Table 5: Components' methods involved in the *Elevated Risk Detected* Use Case

Elevated Risk Detected			
Component	*Base URL*	*Type*	*Signature or Topic*
DSS Dashboard	GUI		
Analytic Workbench	/	GET	/getModel/{modelId}

Table 6: Components' methods involved in the *Ingest Tests and Samples results* Use Case

Ingest Tests and Samples results			
Component	*Base URL*	*Type*	*Signature or Topic*
Data Capture Gateway	/datacapture	POST	/
Data Gateway	/gateway	POST	/publish
Data Gateway		Produce	T1
Data Cleaner		Consume	T1
Data Cleaner		Produce	T2
Data Qualifier		Consume	T2
Data Qualifier		Produce	T3
Data Harmoniser		Consume	T3
Data Harmoniser		Produce	T4
HHR Importer		Consume	T4
Big Data Platform			CREATE

Table 7: Components' methods involved in the *Risk Mitigation/Treatment planning* Use Case

Risk Mitigation/Treatment planning			
Component	*Base URL*	*Type*	*Signature or Topic*

Risk Mitigation/Treatment planning			
Big Data Platform			CREATE
Big Data Platform			READ
Big Data Platform			UPDATE
Big Data Platform			DELETE
Monitoring and Alerting	/	PUT	/ihelp-alert-evaluator/alerttarget/{id}
Monitoring and Alerting	/	DELETE	/ihelp-alert-evaluator/alerttarget/{id}
Monitoring and Alerting	/	GET	/ihelp-alert-evaluator/alertrule
Monitoring and Alerting	/	POST	/ihelp-alert-evaluator/alertrule

Table 8: Components' methods involved in the *Advice Review* Use Case

Advice Review			
Component	*Base URL*	*Type*	*Signature or Topic*
DSS Dashboard	GUI		
Store REST	/	GET	/process
Store REST	/	POST	/process
Big Data Platform			READ
Big Data Platform			UPDATE

Table 9: Components' methods involved in the *Risk Mitigation Delivery* Use Case

Risk Mitigation Delivery			
Component	*Base URL*	*Type*	*Signature or Topic*
Data Gateway	/gateway	POST	/publish
Data Gateway		Produce	T1
Data Cleaner		Consume	T1
Data Cleaner		Produce	T2
Data Qualifier		Consume	T2
Data Qualifier		Produce	T3
Data Harmoniser		Consume	T3
Data Harmoniser		Produce	T4
HHR Importer		Consume	T4
Store REST	/	POST	/process
Big Data Platform			CREATE
Secondary Data Pre-Processor	/v2/	POST	/account/login
Secondary Data Pre-Processor	/v2/	POST	/account/RefreshToken
Secondary Data Pre-Processor	/v2/	GET	/data-provider/answers
Secondary Data Pre-Processor	/v2/	GET	/data-provider/physiological

Risk Mitigation Delivery			
Secondary Data Pre-Processor	/v2/	GET	/data-provider/exercises
Secondary Data Pre-Processor			setMessageAsReceived
Personalised Advisor			sendMessage(message)
Mobile App	GUI		

Table 10: Components' methods involved in the *Monitoring* Use Case

Monitoring			
Component	*Base URL*	*Type*	*Signature or Topic*
Big Data Platform			READ
Monitoring and Alerting	/	POST	/ihelp-alert-data-aggregator/aggregate
Monitoring and Alerting	/	POST	/ihelp-alert-evaluator/evaluate/monitoring
Monitoring and Alerting	/	GET	/ihelp-alert-evaluator/alertrule
Personalised Advisor			generateMessage(vales[])

Table 11: Components' methods involved in the *Advice Follow-up* Use Case

Advice Follow-up			
Component	*Base URL*	*Type*	*Signature or Topic*
DSS Dashboard	GUI		
Store REST	/	GET	/process
Big Data Platform			READ
Monitoring and Alerting	/	GET	/impact-evaluator/getPatientStats
Monitoring and Alerting	/	GET	/impact-evaluator/getMessagesStats
Monitoring and Alerting	/	GET	/impact-evaluator/getReport

Table 12: Components' methods involved in the *Caption of profiling characteristics* Use Case

Caption of profiling characteristics			
Component	*Base URL*	*Type*	*Signature or Topic*
Data Gateway	/gateway	POST	/publish
Data Gateway		Produce	T1
Data Cleaner		Consume	T1
Data Cleaner		Produce	T2

Caption of profiling characteristics			
Data Qualifier		Consume	T2
Data Qualifier		Produce	T3
Data Harmoniser		Consume	T3
Data Harmoniser		Produce	T4
HHR Importer		Consume	T4
Big Data Platform			CREATE
Secondary Data Pre-Processor	/v2/	POST	/account/login
Secondary Data Pre-Processor	/v2/	POST	/account/RefreshToken
Secondary Data Pre-Processor	/v2/	GET	/data-provider/answers
Secondary Data Pre-Processor	/v2/	GET	/data-provider/physiological
Secondary Data Pre-Processor	/v2/	GET	/data-provider/exercises
Mobile App	GUI		

Table 13: Components' methods involved in the *Plan a visit or control* Use Case

Plan a visit or control			
Component	*Base URL*	*Type*	*Signature or Topic*
Data Gateway	/gateway	POST	/publish
Data Gateway		Produce	T1
Data Cleaner		Consume	T1
Data Cleaner		Produce	T2
Data Qualifier		Consume	T2
Data Qualifier		Produce	T3
Data Harmoniser		Consume	T3
Data Harmoniser		Produce	T4
HHR Importer		Consume	T4
DSS Dashboard	GUI		
Store REST	/	POST	/process
Big Data Platform			CREATE
Secondary Data Pre-Processor	/v2/	POST	/account/login
Secondary Data Pre-Processor	/v2/	POST	/account/RefreshToken
Secondary Data Pre-Processor	/v2/	GET	/data-provider/answers
Secondary Data Pre-Processor	/v2/	GET	/data-provider/physiological
Secondary Data Pre-Processor	/v2/	GET	/data-provider/exercises

Plan a visit or control			
Secondary Data Pre-Processor			setMessageAsReceived
Personalised Advisor			sendMessage(message)
Mobile App	GUI		

Table 14: Components' methods involved in the *Data Visualisation* Use Case

Data Visualisation			
Component	*Base URL*	*Type*	*Signature or Topic*
DSS Dashboard	GUI		
Store REST	/	GET	/process
Big Data Platform			READ

Table 15: Components' methods involved in the *Communication of risk* Use Case

Communication of risk			
Component	*Base URL*	*Type*	*Signature or Topic*
Data Gateway	/gateway	POST	/publish
Data Gateway		Produce	T1
Data Cleaner		Consume	T1
Data Cleaner		Produce	T2
Data Qualifier		Consume	T2
Data Qualifier		Produce	T3
Data Harmoniser		Consume	T3
Data Harmoniser		Produce	T4
HHR Importer		Consume	T4
Big Data Platform			CREATE
Secondary Data Pre-Processor	/v2/	POST	/account/login
Secondary Data Pre-Processor	/v2/	POST	/account/RefreshToken
Secondary Data Pre-Processor	/v2/	GET	/data-provider/answers
Secondary Data Pre-Processor	/v2/	GET	/data-provider/physiological
Secondary Data Pre-Processor	/v2/	GET	/data-provider/exercises
Mobile App	GUI		

Table 16: Components' methods involved in the *Social network data extraction* Use Case

Communication of risk			

Communication of risk			
Component	*Base URL*	*Type*	*Signature or Topic*
Social Analyser			elaborateData
Social Analyser			createAlerts
Social Analyser			newApplication
Social Analyser			setSocialMedia
Social Analyser			setLanguages
Social Analyser			setKeywords
Social Analyser			setRules
Social Analyser			startApplication

Table 17: Components' methods involved in the *Patient enrolment using DSS* Use Case

Communication of risk			
Component	*Base URL*	*Type*	*Signature or Topic*
DSS Dashboard	GUI		
Store REST	/	GET	/process
Store REST	/	POST	/process
Big Data Platform			CREATE
Big Data Platform			READ
Big Data Platform			UPDATE

Table 18: Components' methods involved in the *Patient enrolment using Healthentia* Use Case

Communication of risk			
Component	*Base URL*	*Type*	*Signature or Topic*
Data Gateway	/gateway	POST	/publish
Data Gateway		Produce	T1
Data Cleaner		Consume	T1
Data Cleaner		Produce	T2
Data Qualifier		Consume	T2
Data Qualifier		Produce	T3
Data Harmoniser		Consume	T3
Data Harmoniser		Produce	T4
HHR Importer		Consume	T4
Big Data Platform			CREATE
Secondary Data Pre-Processor	/v2/	POST	/account/login
Secondary Data Pre-Processor	/v2/	POST	/account/RefreshToken
Secondary Data Pre-Processor	/v2/	GET	/data-provider/answers
Secondary Data Pre-Processor	/v2/	GET	/data-provider/physiological

Communication of risk			
Secondary Data Pre-Processor	/v2/	GET	/data-provider/exercises
Mobile App	GUI		

Table 19: Components' methods involved in the *Select a preferred risk prediction model* Use Case

Communication of risk			
Component	*Base URL*	*Type*	*Signature or Topic*
DSS Dashboard	GUI		
Analytic Workbench	/	GET	/listModels
Analytic Workbench	/	GET	/getModel/{modelId}

4 Non-Functional specifications

This section incorporates the non-functional specifications useful to address the non-functional requirements reported in D2.3 – "State of the art and requirements analysis III" (M., A., C., +22), which is the final version of the series of deliverables summarizing the work related to the state of the art and finalizing the definition of requirements. The requirement reference code used in D2.3 is stated in square brackets in the following text.

4.1 Security & Privacy

The data providers partners will sign a Data Processing Agreement clearly stating the roles of the different actors, in terms of Data Controller and Data Processor following the GDPR definition (see requirement [T-REQ-T3.3-12]). The (healthcare) organization hosting the study is the Data Controller, whereas the Mobile App owner is the Data Processor.

Furthermore, the data that will be provided as input will be anonymised at the source. The system will refer internally the patient with a unique iHelp identifier (`iHelp_ID`). Patients can already have also an identifier generated by the mobile application (`healthentia_patient_ID`) and their own identifier in pilot's datasets (`pilot_patient_ID`). The iHelp identifier will be automatically generated when clinicians add new patients to the system and the mapping among the identifiers will be registered at the iHelp storage (see requirement [U-REQ-T4.3-01]).

All the external requests must be authorised from the internal Identity Manager (IM) (i.e., Keycloak, introduced in section 2.1.2). Only the authorised users and applications can use the iHelp API. The IM must be configured with a robust mechanism of password reset/restore and a role-based authorisation mechanism.

4.2 Scalability

The solution must support multiple data streaming in input, specifically the Data Ingestion pipeline (see requirement [T-REQ-T3.1-02]). Taking advantage of the asynchronous communication and the dynamic nature of that part of the system, it will be possible to dynamically instantiate more component if needed.

The component that will implement the message bus (i.e. Kafka, described in section 2.1.2), will be configured to allow multiple contemporary pipelines of data ingestion (see requirement [T-REQ-T3.1-03]).

The Big Data Platform will be designed to guarantee the ability of transactions meanwhile handling diverse high rated ingestion workloads, in order to allow the upscaling of the solution (see requirement [T-REQ-T4.4-02]).

4.3 Maintainability

The management of the virtual machine and the hardware provided for the final deployment will be carried out by the actual provider. Every component leader needs the grant to access the virtual machine to configure, install, maintain, or analyse his own component.

The components should be internally designed for easing the updates and the bug fixing procedures (e.g., using SSH connection, management dashboard, custom webpage, etc.).

The developed software will be available as containerised components, in order to ease the deployment and the upscaling of the solution. The components will be managed by the Kubernetes orchestrator, and its dynamic deployment will speed up the manual procedures (see requirement [T-REQ-T3.2-01]). Furthermore, using the orchestrator API the administrator will be able to describe the definition of services to permit the automatic instantiation and deployment for the different deployment scenarios (see requirement [T-REQ-T3.2-03]).

In the next cycle of the incremental development process, the owner of component will analyse the possibility to provide a GUI to let the data provider configure the pipeline for data ingestion directly from his dashboard, as reported in the optional requirement [T-REQ-T3.2-04]).

A UI dashboard should allow the data provider to define the data pipeline, and once this is defined, this should be translated to a REST call to the *orchestrator of choreography* that will deploy the data pipeline.

4.4 Reliability

The internal storage of the platform will be able to keep correlated, annotated and interoperable data as linked ontologies, allowing the storage of the semantic facts and the corresponding data schema model. If those data are produced and consumed by the same component, it should be also possible to store them internally, in order to not overload the Big DataPlatform used from the whole solution (see requirement [T-REQ-T3.4-17]).

The Big DataPlatform will guarantee a stable and secured ([T-REQ-T3.4-18]) connection towards the other components, allowing the connection from a pool of clients (see requirement [T-REQ-T4.4-02]).

4.5 Usability

The specifications about the usability, user experience, look & feel and other user-related topics have been addressed in the T2.4 – “User Centred Design” and further detailed in the D2.8 – “User Centric Design I” (M., N., C., +22) and D2.9 – User Centric Design II deliverables.

4.6 Minimum and recommended infrastructure requirements

The recommended system requirements for the different pilots’ installations of the iHelp infrastructure, resulting from the individual iHelp components requirements and the configuration of each pilot iHelp infrastructure, and available in chapter 8 of D2.5 (M, P, A, +22), are reported in Table 20:

Table 20: iHelp infrastructure system requirements for different pilots’ installations

Pilot	On premises	VMs No.	Cores No.	RAM (GB)	Storage (GB)
UNIMAN	✓	3	16cores per VM	32GB per VM	150GB per VM
FPG	✓	3	16cores per VM	32GB per VM	150GB per VM
HDM		3	16cores per VM	32GB per VM	150GB per VM
MUP	✓	4	16cores per VM	32GB per VM	150GB per VM
TMU	✓	3	16cores per VM	32GB per VM	150GB per VM

Moreover, it is worth to mention that the iHelp platform, when deployed and configured at the pilots’ premises, should require an Ubuntu 20.04 or Ubuntu 22.04 installation, while a Debian 11 Operating

System (OS) can be also an alternative solution. Finally, the Docker and Docker-Compose tools should have also been installed and configured on the provided VMs by the pilot partners to further facilitate the utilization of the iHelp components.

4.7 Other

The web application designed for Health Care Professionals and Model Builders will be usable through a modern web browser, such as:

- Google Chrome (v80+)
- Safari (13+)
- Firefox (v72+)
- Edge (80+)

The mobile application for individuals and patients will be usable through a smartphone:

- Android (v6+)
- iOS (v10+)

5 Conclusions

This deliverable reports the final version of the functional and non-functional specifications of the overall iHelp platform, aimed at specifying the interfaces between the iHelp components and delegating details on internal functions of the components and processing to the respective deliverables. The specifications reported in the previous version of this deliverable, D2.6 – "Functional and Non-Functional Specifications I" (Mel, Pan, 21), have been revised taking into account new and updated requirements coming from D2.3 – "State of the art and requirements analysis III" (M., A., C., + 22) and the final release of the iHelp architecture in D2.5 – "Conceptual model and reference architecture II" (M., P., A., +22).

Functional specifications have been enhanced thanks to a more detailed knowledge of the components, which has been continuously increased after the release of D2.6 – "Functional and Non-Functional Specifications I". Specifically, the APIs provided by the iHelp components via REST interfaces were documented using Swagger/Open API: 6 Swagger files are available in GitHub. All documented components are described in deliverable D2.5 – "Conceptual model and reference architecture II" (M., P., A., +22) and satisfy the user and technical requirements included in deliverable D2.3 – "State of the art and requirements analysis III" (M., A., C., + 22).

Twenty-eight sequence diagrams are presented in this document aiming at modelling how the components interact to carry out the functions described by the eighteen use cases included in D2.3 – "State of the art and requirements analysis III" (M., A., C., + 22) and refined in D2.5 – "Conceptual model and reference architecture II" (M., P., A., +22) and in the present document, and the messages exchanged among the components. A traceability matrix is also included to better detailing which module of each component is impacted by which use case scenario, in order to satisfy the behaviours modelled in the sequence diagrams.

The non-functional specifications aim at meeting the non-functional requirements reported in D2.3 – "State of the art and requirements analysis III" (M., A., C., + 22) have been also provided.

Bibliography

O. Garcia Perales, J. G. López, A. Ramiro and F. Picioroaga, “D4.4 – Model Library Implementation and Recalibration of Adaptive Models I”, April 2022.

P. Kranas, P. Martinez, J. Pereira, L. M. Garcia, G. Manias, E. Kouremenou, M. Patiño and A. Azqueta, “D3.3 – Primary data capture and ingestion I”, October 2021.

P. Kranas, P. Martínez, J. Pereira, L. M. Garcia and J. Roldan, “D4.9 - Big data platform and knowledge management system I”, December 2021.

G. Manias, U. Wajid, A. Dalianis, E. Kouremenou and A. Azqueta, “D3.7 – Standardisation and Quality Assurance of Heterogeneous Data I”, October 2021.

G. Marinos et al, “D2.1 – State of the art and requirements analysis I”, June 2021.

G. Marinos et al, “D2.2 – State of the art and requirements analysis II”, December 2021.

G. Marinos et al, “D2.2 – State of the art and requirements analysis III”, June 2022.

N. Mehandjiev, P. Nikolova and N. Khan, “D 2.9 – User-Centric Design II”, August 2022.

F. Melillo and C. Pandolfo, “D2.6 – Functional and Non-Functional Specifications I”, October 2021.

F. Melillo, L. Pucci, A. Azqueta, A. Ramiro, P. Kranas, D. Kirova, G. Manias, G. Giotis, H. op den Akker, K. Filipov, O. Garcia Perales and U. Rashid, “D2.5 – Conceptual model and reference architecture II”, June 2022.

H. op den Akker, T. Beinema, A. Pnevmatikakis, S. Spanos, S. Kanavos, A. Pantazidis, V. Tzironis, K. Kostopoulou, S. Garcia Torrens and A. Damiani, “D5.6 – Delivery Mechanisms for Personalised Healthcare and Real-Time Feedback I”, April 2022.

A. Pnevmatikakis, S. Spanos, A. Perikleous, H. Op den Akker, P. Kranas, U. Wajid and C. Orton, “D3.5 – Secondary data capture and interoperability I”, October 2021.

U. Rashid, U. Wajid, O. Garcia, L. Gustafsson, F. Andersson and T. Tomson, “D5.8 – Social Analytics for the Study of Societal Factors and Policy Making I”, April 2022.

List of Acronyms

AI	Artificial Intelligence
API	Application Programming Interface
CA	Consortium Agreement
D	Deliverable
DoA	Description of Action
DSS	Decision Support System
EU	European Union
FPG	Fondazione Policlinico Universitario Agostino Gemelli
GDPR	General Data Protection Regulation
HCP	Health Care Professional
HDM	Hospital de Denia-MarinaSalud
HHR	Holistic Health Record
IAM	Identity and Access Manager
IM	Identity Manager
ID	Identifier
JDBC	Java Data Base Connectivity
JWT	JSON Web Token
M	Month
MUP	Medical University Plovdiv
PC	Pancreatic Cancer
REST	Representational State Transfer
T	Task
TMU	Taipei Medical University
UC	Use Case
UML	Unified Modeling Language
UNIMAN	University of Manchester
UPM	Universidad Politécnica de Madrid
UPRC	University of Piraeus Research Centre
WP	Work Package