Article

EREBOTS: Privacy-Compliant Agent-Based Platform for Multi-Scenario Personalized Health-Assistant Chatbots

Davide Calvaresi [1]*, Jean-Paul Calbimonte [1], Enrico Siboni [1], Stefan Eggenschwiler [1], Gaetano Manzo [1], Roger Hilfiker [1], and Michael Schumacher [1]

[1] University of Applied Sciences and Arts Western Switzerland HES-SO; name.surname@hevs.ch
* Correspondence: davide.calvaresi@hevs.ch; Institute of Information Systems HES-SO Valais-Wallis, TechnoPole 3, CH-3960 Sierre, Switzerland.

Version March 4, 2021 submitted to Electronics

Abstract: *Context.* Asynchronous messaging is increasingly used to support human-machine interactions, generally implemented through chatbots. Such virtual entities assist the users in activities of different kinds (e.g., work, leisure, and health-related) and are becoming ingrained into humans' habits due to factors including *(i)* the availability of mobile devices such as smartphones and tablets, *(ii)* the increasingly engaging nature of chatbot interactions, *(iii)* the release of dedicated APIs from messaging platforms, and *(iv)* increasingly complex AI-based mechanisms to power the bots' *behaviors*. Nevertheless, most of the modern chatbots rely on state machines (implementing conversational rules) and one-fits-all approaches, neglecting personalization, data-stream privacy management, multi-topic management/interconnection, and multi-modal interactions. *Objective.* This work addresses the challenges above through an agent-based framework for chatbot development named EREBOTS. *Methods.* The foundations of the framework are based on the implementation of *(i)* multi-front-end connectors and interfaces (i.e., Telegram, dedicated App & web interface); *(ii)* enabling the configuration of multi-scenarios behaviors (i.e., preventive physical conditioning, smoking-cessation, and support for breast-cancer survivors), *(iii)* online learning, *(iv)* personalized conversations and recommendations (i.e., mood boost, anti-craving persuasion, and balance-preserving physical exercises), and *(v)* responsive multi-device monitoring interface (i.e., doctor and admin). *Results.* EREBOTS has been tested in the context of physical-balance preservation in social confinement times (due to the ongoing pandemic). Thirteen individuals characterized by diverse age, gender, and country distribution have actively participated in the experimentation, reporting advancements in the physical balance and overall satisfaction of the interaction and exercises' variety they have been proposed.

Keywords: chatbot, multi-agent systems, personalized virtual assistant, privacy agents, eHealth, conversational agent.

---

1. Introduction

Intelligent systems constitute the backbone of increasingly popular services and applications used to support people in several activities. Such applications have the ability to assist humans through multi-modal interactions, including text, buttons, vocal, video, and gesture-based communication. Siri[1], Cortana[2], and Alexa[3] are among the most known at a commercial level and lead customers' trends and hypes. Although such virtual assistants heavily rely on vocal interactions [1,2], there are several cases where more discrete and asynchronous chat-like communications are still preferred. Chatbots are an example of intelligent systems

---

[1] https://www.apple.com/siri/
[2] https://www.microsoft.com/en-us/cortana
[3] https://developer.amazon.com/en-US/alexa

relying on interactions mostly menu/text-based. In particular, a chatbot is a computer program able to entertain a natural language-based conversation with a human. While the first ancestors of conversational agents date back to the 60s (e.g., ELIZA [3]), the features and capabilities of chatbots have experienced a tremendous improvement relatively recently. Several solutions adopt Natural language processing (NLP) coupled with AI-based mechanisms to build/elaborate the chatbots' knowledge base, which generally consists of a collection of dialogue management rules, behaviors, background, aggregated data, settings, and a collection of techniques for data manipulation. Among the factors contributing to this increasing adoption, we can mention anywhere/anytime availability, immediate response, confidentiality, social acceptance, and massive scalability. Thanks to these factors, chatbots have shown to be effective in a wide range of domains, particularly for motivational (e.g., social network campaigns [4]) and support (e.g., customer management [5], eHealth [6], and assisted-living scenarios [7]).

In the healthcare domain, chatbots leveraging on tailored support and social aspects can be of great support to foster behavioral change (e.g., smoking cessation) [4,6,8], monitoring of chronic health conditions [9], primary care [10], etc. However, modern chatbots are still affected by significant limitations such as inadequate personalization, lack of real-time monitoring, reporting and customization for medical personnel, lack of mechanisms to integrate communities of chatbots, limited knowledge sharing capabilities, and the impossibility of seamlessly deploying multi-domain campaigns within the same framework. These limitations are linked to the predominantly rigid architectures proposed in most existing approaches. These rely on very specific scenarios translated into chatbot logics, which have to be reprogrammed every time a new scenario arrives. This raises the costs of modifying a chatbot's behavior and prevents healthcare professionals from adapting it to certain situations. Moreover, most chatbot solutions rely on monolithic and centralized data management strategies, making it hard to comply with privacy regulations (e.g., GDPR [11]). The sensitive nature of data collected through chatbot interactions makes it necessary to shift the control of personal data towards the users themselves, empowering them in the process.

This paper tackles the above-mentioned limitations through an agent-based framework (named EREBOTS), which enables the configuration and deployment of personalized chatbots to support users in multi-topic and multi-campaign behavioral change programs. Examples include conversational agents coaching people fighting chronic diseases, addictions, and other health issues, leading to decreased life quality. In particular, the **contribution** is five-folded:

- **Multi-scenario agent-based chatbot framework:** In EREBOTS, it is possible to combine several context-dependent behaviors that can be encapsulated in dedicated *story lines*, which can be modeled as isolated or interconnected scenarios. These behaviors are enacted by a network of *user agents*, *doctor agents*, and orchestrated through *gateway agents*.
- **User personalization:** User agents build a model of the user profile, his/her preferences, history, goals, and aggregated information. With this model, the user agents are able to tailor behaviors and provide a personalized experience.
- **Healthcare personnel control and monitoring:** Medical doctors and healthcare providers have the possibility of defining possible goals, configure self-assessment interactions, or customize the types of activities proposed to patients/participants. Moreover, they can monitor users' profiles with detailed analytic describing their behaviors and aggregated trends.
- **Privacy and ethics compliance:** In EREBOTS, all the sensitive/personal information are solely under the control of the user, who can make any decisions concerning storage and sharing of her information. Through the Pryv. platform [12] integrated into EREBOTS, users may configure fine-grained access control or even entirely remove their data if they decide so.
- **Multi-campaign implementation and testing:** EREBOTS has been employed and tested in scenarios such as smoking cessation and balance enhancement exercises (physical rehabilitation) for older adults during social confinement (due to COVID-19 restrictions).

The rest of the paper is organized as follows. Section 2 presents the state of the art and elicits the open challenges. Section 3 details the framework, its components, behaviors, and interfaces. Section 4 describes the test-bed scenario and elaborates on the test results. Section 5 relates and discusses the developed platform and the open challenges. Finally, Section 6 concludes the paper.

2. State of the Art

The contributions presented in this work lay at the intersection of different disciplines, including Human-Machine Interface (HMI), Quality of Experience (QoE), intelligent personalized systems (i.e., multi-agent systems), and persuasive healthcare/assistive technologies.

2.1. HMI & Chatbots

Nowadays, the market can count a plethora of applications providing conversational services. However, only a few of them are able to keep pace with the latest trends. In particular, platforms such as Telegram, Facebook, (and slowly WhatsApp) have released APIs to develop chatbots. Initially, such functionalities were mostly used in early prototypes and niche-application domains such as e-commerce and customer care support. Recently, several chatbots-based services and frameworks emerged, fostering further developments in the area. Among these, we can mention:

- Amazon Lex: it supports the development of chatbots providing natural language understanding and automatic speech recognition [13].
- Dialogflow: it provides a framework aiming at understanding human conversations relying on Google's machine learning techniques [14].
- Microsoft Bot Framework: it is a tool-set including APIs for text and speech analysis [15].
- SAP Conversational AI: based on SAP's technology platform, it enables users to build and monitor intelligent chatbots, as well as to automate tasks and workflow [16].
- Rasa Open Source: it is a machine learning framework that allows the automation of text and voice-based chatbot assistants [17].

These frameworks tackle primarily natural language and speech processing, providing little support to the management of conversation coordination, user profiling, and user experience. Beyond these commercial solutions, further research has been performed regarding human-computer interaction approaches that enrich chatbots with social characteristics in order to cope with frustration and dissatisfaction [18]. The human factor in this type of interaction is not negligible, given the differences in perception [19] and emotional state [20], which can lead to entirely different paradigms for designing a conversational agent and evaluating it [21]. Moreover, despite the increasingly strict regulations in the matter of personal data usage [22], the services mentioned above often collide with required confidentiality and privacy restrictions (especially for health-related programs [23]). Users interacting with chatbots have little or no control over personal and sensitive data exchanged or processed within the context of the conversational agent activities.

2.2. Quality of Experience

In the 2000s, QoE focused on bridging the gap between technical quality metrics (i.e., QoS) and the user's subjective perception of the service quality [24]. Usually, QoE is employed to assess a service beyond its technical aspects. When human users are involved, the system's performance is always perceived subjectively due to several factors [25]. For example, we can name three categories:

(i) human factors – such as user personality, expertise, health condition (visual acuity, auditory capacity, etc.);
(ii) context factors – such as the context in which a user is consuming a given service (e.g., alone, with friends, on the way to work, etc.); and
(iii) system factors – such as a system's features characterizing the service provided (e.g., video resolution, sound quality, response rate, Natural language processing quality, etc.).

QoE enables to assess the end-user satisfaction comprehensively. Recent studies map QoE to multi-agent systems (MAS). In particular, QoE comes handy when modeling users' satisfaction, expectations, and the will to maximize their objective with intelligent agents [26]. Each user can be bounded with a personal agent representing his/her context & preferences and acts on his/her behalf [27].

2.3. Multi-Agent Systems & Chatbots

Model-wise, chatbots and agents have remarkable overlaps. In literature, they can be considered completely matching (in term of functionalities, knowledge, behaviors, and user mapping) [4,6] or modeling the chatbot as an interface for a more *complex, intelligent*, and possibly distributed system [28,29]. Bentivoglio et al. [30] embody the combination of chatbot - agent(s) as a stimulus-reply state automaton and a goal-driven probabilistic agent (defined as a Partially Observable Markov Decision Process). The user can stimulate the chatbot in a predefined manner (i.e., via a menu) or via natural language. During the entire conversation, the agent relates the possible actions to two main goals: *(i)* an *immediate goal* - achievable in a single dialog step, *(ii)* a *global goal* - to be achieved by the end of the conversation. Moreover, elements of pragmatics can be added in the dialogue description, thus enhancing the adherence of the chatbot's behavior to the user mood and the overall interaction [31].

Solutions exploiting agent-based chatbots can model and, in turn, implement better responses to environmental stimuli coupled with the human - virtual information flow. Agent-based chatbots can push the interactions and capabilities far beyond the conventional (mainly procedural/static) interactions characterizing chatbot employed in a plethora of application domains (i.e., retail [32], tourism [33], etc.) For example, Żytniewski [34] studied agent-based chatbots as a bridge between users and IT systems in business processes and management of the organization knowledge. Alencar and Netto [35] proposed an approach to improve the cooperation among students and learning-institution. In particular, they realized an *Assistant Tutor* agent responsible for the *(i)* question collection, *(ii)* activity monitor, and *(iii)* student interaction a virtual learning environment (i.e., Moodle). Hettige and Karunananda [36] proposed Octopus, a multi-agent assistant chatbot using the Sinhala language and aiming at automatizing a limited amount of tasks such as opening/closing applications, search in text, and executing generic commands. Finally, Calvaresi et al. [37] proposed a framework to realize agent-based chatbots for smoking cessation purposes. While they have outlined a *multi*-agent design of their solution, they have implemented *single*-agent framework and highlighted the envisioned gaps among the two solutions.

2.4. Chatbots in assistive & eHealth scenarios

In the context of eHealth and assistive application scenarios, well-known properties such as anonymity, asynchronicity, personalization, scalability, authentication, and consumability represent an inherited plus for the applications leveraging on chatbot technologies [38]. In this context, the most relevant application scenario are chronic illness attention [39,40], interviews [41,42], counseling [43,44], chronic health conditions monitoring [45,46], medication adherence [47,48], self-care [49], promoting healthy behavior [7], counseling and social therapy [50], and primary care [40]. According to Pereira and Diaz [38], in the context of behavioral change, chatbots are employed in a three-dimensional space considering illnesses (or health issues), competencies (e.g., cognition, behavior, monitoring), and enablers (e.g., anonymity, asynchronicity, scalability). From their analysis, the main categories characterizing the illness dimension are organized in Figure 1. Besides the *specific* contribution, the solutions elaborated in [38] are generally **not usable in mobile phones**, mostly due to **browser-plugin requirements or assumption of large-screen availability**. Such a drawback hampers the usability, losing the chatbots' inherited advantages (particularly timeliness, pervasiveness, and accessibility).

Among the use-cases where chatbots have been employed, we can cite smoking cessation campaigns, where the need for intervention and support, especially via social networks, has been reported [51–53]. Tweet2Quit [54] is an example of such bots, focusing on daily automated twitter-delivered communications to small and private self-help groups to encourage discussions on smoking cessation. However, the evidence is not conclusive and does not yet show the efficacy of this approach. Regarding chronic diseases, Brixey et al. [55] proposed a Facebook-based chatbot to deliver sexual health information on HIV/AIDS to young adult. Similarly, deployed on the Telegram platform, Vita et al. [56] designed a chatbot to improve people's engagement in living with HIV, assisting them in booking visits and managing the theory.

Other application scenarios for chatbots include food counseling, as in Fadhil et al. [7], who present a chatbot fostering a sustainable and healthy lifestyle and preventing weight gain in adult individuals. Ni et al. [10] focused on primary care patient intake, presenting a chatbot as a proxy between patients and physicians,

Figure 1. Scheme of illnesses categories that employ chatbots, adapted from [38].

collecting their chief complaints in natural language, then reported to the doctors for further analysis. Concerning dietary & food counseling, the contributions span from conversational agents for assisting users in the kitchen (exploiting Watson to orchestrate conversation) [57] to chatbots assisting young adults with food allergies to find information about restaurants, sharing concerns, and ask for further information via existing messaging apps (i.e., Messenger) [58]. Ghandeharioun et al. [59] proposed a chatbot sampling *"emotions"* and responding with appropriated empathy. The authors tried to grasp the meaning of emotional intelligence in the context of a chatbot, touching both objective and emotional topics and investigating the chatbot's influence on the users' behavior. Finally, in [60] a serious game was presented, involving medical students with the objective of training them in patient-centered medical interviews, exploiting agent-based chatbots.

2.5. Opportunities and open challenges

Elaborating on the evidence highlighted by the existing studies, chatbots operating in assistive/healthcare scenarios have great potential to: *(i)* disseminate health information and coaching instructions & suggestions; *(ii)* profile users to provide personalized information and advice; *(iii)* motivate and induce positive behavioral change; *(iv)* support persuasive strategies for adherence and self-efficacy. Nevertheless, the following open challenges/issues need to be addressed:

- **C1 Social A2A (Agent-to-Agent):** While chatbots have been mainly employed in social campaigns, the *social capabilities* among the bots (i.e., to relate/extend/complete information) have yet to be fully exploited.
- **C2 Run-time healthcare supervision:** Mental & physical wellness and nutritional & metabolic disorders are areas that can vastly benefit from employing chatbots to attain behavior change. Nevertheless, physicians consider unsafe to release *unsupervised* autonomous chatbots operating in safety-critical scenarios [61].
- **C3 Evolving models & behaviors:** Chatbots can model the users quite comprehensively. However, the sociological dynamics and implications can quickly change, and current solutions cannot model nor properly embed evolving behaviors in the complex dynamics of current frameworks.
- **C4 Multi-stakeholder personalization:** Chatbot are pervading increasingly complex healthcare applications. However, current solutions do not provide sufficient personalization for the diverse stakeholders' roles (i.e., caregivers, physicians, or relatives [37]).

C5 Users' QoE: The user is central in chatbot applications. Nevertheless, mechanisms to *periodically* collect, elaborate, and understand users' feedback on their experience are missing [62].

C6 Dynamic update mechanisms: The repetitiveness of the solutions and/or functionalities suggested by the chatbots (usually due to static state-machines and the lack of run-time updating mechanisms) can cause users to relapse and abandon the application.

C7 Semantics & Terminology: Often, the messages sent by the chatbot are predefined. However, due to the diversity of the stakeholders in healthcare scenarios, the terminology and related sentence formulation should be formulated dynamically (i.e., standardization vs. explanation).

C8 Delegation: Chatbots can replace man-power in dealing with automated and repetitive tasks. However, the criteria for delegating a task (computational and interaction-wise) to a chatbot need to be defined [63].

C9 Privacy compliance: While the chatbots' interactions are mostly visible to the user, what occurs in the back-end is usually not as clear/transparent. In the best-case scenario, data management and visibility are described in human-made informative documentation, where the actual match with the system dynamics cannot be verified.

Tackling such challenges is crucial for a society experiencing a remarkable increase in awareness about people's health. Indeed, healthcare and eHealth systems are facing the strain of a significant demand for user (patient) empowerment –implying the need for new logics, architectures, dynamics, and interfaces [4,37,64]. Employing MAS models and techniques to realize chatbot is promising, yet, in an early stage (see the open challenges). Above all, integrating the capabilities of conversational agents within the MAS dynamics has not been fully exploited.

3. The EREBOTS framework

The design of EREBOTS serves as a base to overcome the challenges mentioned above. Figure 2 schematizes the underlying architecture of EREBOTS, and Figure 3 depicts the main components per container and their interactions.

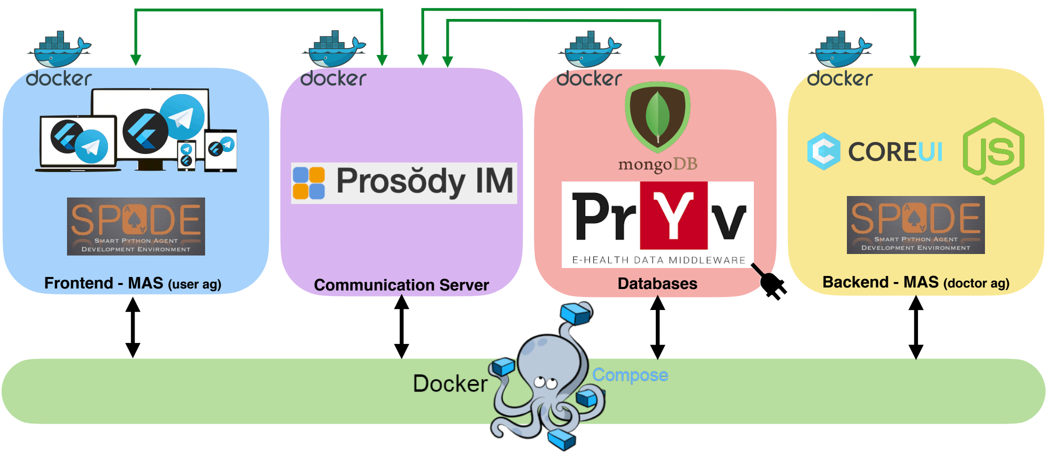


Figure 2. EREBOTS architecture and interactions containerized via Docker.

The framework comprises four main components: Database(s), Communication Server, MAS back-end for the doctor agents, MAS for the user agents & front-end. Each of these components is deployed on a dedicated container and managed through Docker Compose.

- The *Database* component encloses two different databases: (i) *MongoDB*, used as centralized storage only for non-personal data. In particular, it stores the user's messenger service chat ID (e.g., Telegram) and the user-specific endpoint token for the personal data store, and (ii) *Pryv*[4], which is a platform enabling

[4] https://www.pryv.com/

privacy regulation-compliant, stream-based personal data collection, and privacy management. Once a user has registered an account, the user can provide consent to external applications, which then can access and store specified data. EREBOTS uses an instance of Pryv to persist the user's chat history and all personal data (e.g., age, name, and scenario-specific data). Employing Pryv, users gain exclusive control of their data, thus being able to revoke the consent at any point, disabling EREBOT access to it, and, if necessary, fully removing any stored piece of information.

- The *Communication server* acts as message space for the inter-agent communication within the MAS. It uses a Prosody[5] XMPP server instance where each agent embodies a registered user. An agent can broadcast messages to all agents (in the form of a multi-user chat) or directly message a specific agent (in the form of peer-to-peer sessions).
- The *Back-end* relies on the SPADE framework [65] to instantiate and interconnect virtual agents. In particular, it endows the doctor agent, which serves the campaign-related functionalities and bridges them with the underlying system's dynamics. Moreover, the doctor agent exposes a web application allowing the medical personnel in charge of the campaign to manage storylines (general or personalized therapies) and overview user treatments adherence/results.
- The *Front-end* component is in charge of managing the users' connections and their messages from the chat platform(s). Although extensible to other messaging systems, the framework currently supports the following communication interfaces: (i) *Telegram*[6]*:* a widely used free messaging application for mobile phones released in 2013, offering desktop applications for PC, Mac, and Linux. Since 2015, Telegram has enabled the development of chatbots with a dedicated bot API. (ii) *HemerApp:* a dedicated front-end based on Flutter[7], a framework for native multi-platform development. Therefore, the HemerApp can be used on iOS, Android, or web.

While HemerApp allows a direct connection with the MAS (i.e., SPADES), all messages using Telegram have to *pass through* dedicated Telegram APIs. This required the realization of a *gateway agent*. Moreover, such an agent handles the initial user communication (i.e., registration and *user agent* creation) for both interfaces. As of today, the two interfaces can coexist, although only one is allowed within a given campaign.

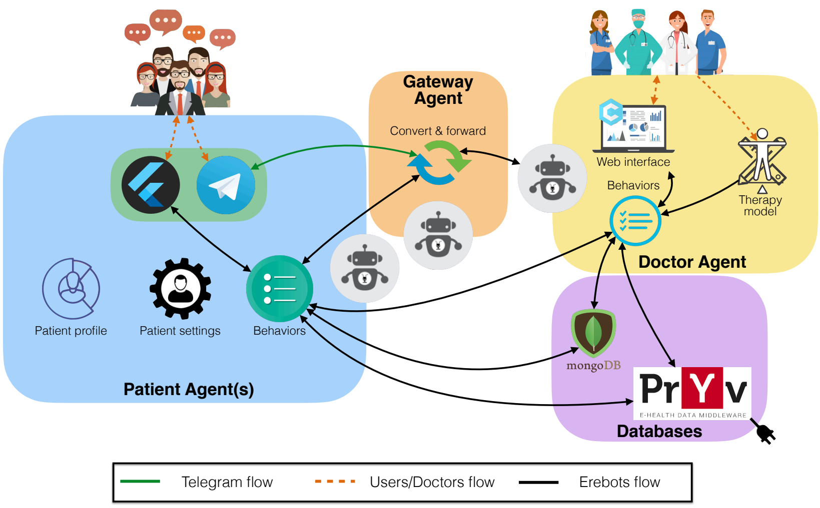


Figure 3. Main components, data stream, and agents' interactions of the EREBOTS architecture.

The user data model can be considered hybrid (i.e., storing information coming from Telegram and HemerApp in MongoDB and Pryv contextually).

---

5 https://prosody.im/
6 https://telegram.org/
7 https://flutter.dev/

If Telegram is the front-end, the user data persisted in MongoDB are: Telegram ID, first and last names, interaction language, last user's interaction, Pryv endpoint – to read from and write to events in the user's Pryv data streams –, and a boolean variable related to user registration; and those stored in Pryv are: age, sex and any other data relevant for the given campaign (see Listing 1). The messages exchanged between EREBOTS and the user are stored on the Telegram platform.

Listing 1. User Model: Telegram front-end.

```
class User(BasicUser):
    """Actual model class for user data stored in mongo_db"""

    telegram_id = StringField()
    first_name = StringField(required=True)
    last_name = StringField()
    language = StringField(choices=Language.values())
    last_interaction = DateTimeField()
    pryv_endpoint = StringField()
    registration_completed = BooleanField(default=False)

class PryvStoredData(ValuesMixin):
    """Enum class with all permissions requested to the user"""

    AGE = ("age", "Age Range", AccessLevel.MANAGE)
    SEX = ("sex", "Sex", AccessLevel.MANAGE)
    FAVOURITE_SPORT_DAYS = ("favourite_sport_days", "Favourite Sport Days", AccessLevel.MANAGE)
    GOALS = ("goals", "Goal IDs", AccessLevel.MANAGE)
    CURRENT_QUESTION = ("current_question", "Current Question ID", AccessLevel.MANAGE)
    CURRENT_QUESTION_ANSWER = ("current_question_answer", "Current Question Answer", AccessLevel.MANAGE)
    SPORT_SESSIONS = ("sport_sessions", "Sport Session", AccessLevel.MANAGE)
```

If HemerApp is the selected front-end, the data model (see Listing 2) differs from the model shown in Listing 1 as follows: only the user's chat id, the Pryv endpoint, and the boolean flag related to registration are stored in the local MongoDB instance. All other user-related personal data - including all the exchanged messages - is persisted in the form of a Pryv data stream and is thus under the sole user's control.

Listing 2. User Model: HemerApp front-end.

```
class User(BasicUser):
    """Actual model class for user data stored in mongo_db"""

    telegram_id = StringField()
    custom_chat_id = StringField()
    pryv_endpoint = StringField()
    registration_completed = BooleanField(default=False)

class PryvStoredData(ValuesMixin):
    """Enum class with all permissions requested to the user"""

    FIRST_NAME = ("covid19_first_name", "[Covid19Project] First Name", AccessLevel.MANAGE)
    LANGUAGE = ("covid19_language", "[Covid19Project] Language", AccessLevel.MANAGE)
    AGE = ("covid19_age", "[Covid19Project] Age Range", AccessLevel.MANAGE)
    SEX = ("covid19_sex", "[Covid19Project] Sex", AccessLevel.MANAGE)
    FAVOURITE_SPORT_DAYS = ("covid19_favourite_sport_days", "[Covid19Project] Favourite Sport Days",
    ↪  AccessLevel.MANAGE)
    GOALS = ("covid19_goals", "[Covid19Project] Goal IDs", AccessLevel.MANAGE)
    CURRENT_QUESTION = ("covid19_current_question", "[Covid19Project] Current Question ID",
    ↪  AccessLevel.MANAGE)
    CURRENT_QUESTION_ANSWER = (
        "covid19_current_question_answer", "[Covid19Project] Current Question Answer", AccessLevel.MANAGE
    )
    SPORT_SESSIONS = ("covid19_sport_sessions", "[Covid19Project] Sport Session", AccessLevel.MANAGE)
    CHAT_MESSAGES = ("covid19_chat_messages", "[Covid19Project] Exchanged Chat Messages",
    ↪  AccessLevel.MANAGE)
```

Listing 3 shows a snippet of the implementation of the user object. In particular, *(i)* the *pryv_endpoint* (persisted in the local MongoDB instance) is implemented as shown in lines 31-38; *(ii)* the *age* is stored as a data stream in Pryv and implemented as shown in lines 22-29.
Lines 4-11 show how to read the Pryv properties. Specifically, it is a parameterized HTTP GET request sent to the Pryv endpoint. The additional parameters include the ID of the stream (in this case *covid19_age* – see Listing 2) and the desired response limit (i.e., how many stream elements are returned). Writing the stream is

actualized as an HTTP POST request. The ID of the stream is required, as well as the new value to be added to the stream (Lines 13-20).

Listing 3. Snippet of the user object implementation.

```
1 class MongoDBAndPryvUser(AbstractUser, MongoDBUserMixin, MongoDBObjectWithIDMixin):
2     """Actual implementation of AbstractUser for MongoDB and Pryv hybrid"""
3
4     def _access_pryv_last_value_of(self, stream_id: str) -> Optional[str]:
5         """Utility method to access the last value of a Pryv stream"""
6         user_endpoint_with_token = self._user_mongodb_obj.pryv_endpoint
7         if user_endpoint_with_token:
8             stream_events = self._pryv_api.get_events(user_endpoint_with_token, streams=[stream_id],
              ↪  limit=1)
9             return stream_events[0].content if stream_events else None
10        else:
11            return None
12
13    def _set_pryv_new_value_for(self, stream_id: str, new_value: str):
14        """Utility method to set a new event in a Pryv stream"""
15        user_endpoint_with_token = self._user_mongodb_obj.pryv_endpoint
16        if user_endpoint_with_token:
17            self._pryv_api.create_event(user_endpoint_with_token, [stream_id], new_value)
18        else:
19            logger.warning(f"Not setting value {new_value} for {stream_id},"
20                           f" because the user has not a Pryv endpoint set.")
21
22    @property
23    def age(self) -> Optional[AgeField]:
24        age = self._access_pryv_last_value_of(PryvStoredData.AGE.value[0])
25        return AgeField(age) if age else None
26
27    @age.setter
28    def age(self, new_value: AgeField):
29        self._set_pryv_new_value_for(PryvStoredData.AGE.value[0], new_value.value)
30
31    @property
32    def pryv_endpoint(self) -> Optional[str]:
33        return self._user_mongodb_obj.pryv_endpoint
34
35    @pryv_endpoint.setter
36    def pryv_endpoint(self, new_value: str):
37        self._user_mongodb_obj.pryv_endpoint = new_value
38        self._user_mongodb_obj.save()
```

Listing 4 shows an extract of the log generated by a communication occurring via the Telegram front-end and directed to the gateway agent. The process starts by receiving the first message from a given Telegram user-name, i.e., “John” (Line 1). It triggers the gateway agent to search for the user in its cache (Line 2). In this extract, the research is unsuccessful. Thus, the Gateway Agent contacts the Doctor Agent, who queries the local MongoDB instance. (Line 3). Such a mechanism is necessary due to the availability of multiple user interfaces (i.e., Telegram and HemerApp), research still unsuccessful. Thus, the Doctor Agent creates a new MongoDB object for John (Line 4). In turn, the Gateway Agent creates the associated User Agent, and the underlying MAS framework (i.e., SPADE) registers a new user in the XMPP server and links it to the user agent (Lines 5-7). Once the creation concludes successfully, the message triggering the registration is forwarded to the proper User Agent (Lines 8-9), which continues the user’s profiling as instructed (Lines 10-13).

Figure 4(a) shows the results of the user registration process into MongoDB. It is worth to notice, that those profiles who did not complete the registration do not have generated the Pryv endpoint. Figure 4(b) shows the streams persisted in Pryv as results of the user registration performed with HemerApp.

Listing 4. Initial user-to-EREBOTS communication.

```
{"log":"INFO:common.telegram.bot: received_known_command: `/start` from `John` with ID `1725`\n"}
{"log":"INFO:common.agent.agents.abstract_gateway_agent:[covidphysio_telegram_gateway_agent] I haven't
↪ cached information about John Doe \n"}
{"log":"INFO:common.agent.agents.abstract_gateway_agent:[covidphysio_telegram_gateway_agent] Asking
↪ covidphysio_doctor_agent@prosody.localhost for information...\n"}
{"log":"INFO:common.agent.agents.abstract_gateway_agent:[covidphysio_telegram_gateway_agent] Received
↪ data for John Doe: {\"_id\": {\"$oid\": \"5fcdea5ab08efe960ba18f28\"}, \"first_name\": \"John\",
↪ \"last_name\": \"Doe\", \"language\": \"LANGUAGE_ENGLISH\", \"last_interaction\": {\"$date\":
↪ -62135596800000}, \"registration_completed\": false}\n"}
{"log":"INFO:covidphysio.common.agent.agents.abstract_covidphysio_gateway_agent:
↪ [covidphysio_telegram_gateway_agent] Creating the UserAgent with jid
↪ john-5fcdea5ab08efe960ba18f28@prosody.localhost\n"}
{"log":"INFO:spade.Agent:Agent john-5fcdea5ab08efe960ba18f28@prosody.localhost connected and
↪ authenticated.\n"}
{"log":"INFO:covidphysio.common.agent.agents.user.agent:[john-5fcdea5ab08efe960ba18f28] UserAgent
↪ started.\n"}
{"log":"INFO:covidphysio.common.agent.agents.abstract_covidphysio_gateway_agent:
↪ [covidphysio_telegram_gateway_agent] Forwarding the messaging platform message to UserAgent
↪ john-5fcdea5ab08efe960ba18f28@prosody.localhost.\n"}
{"log":"INFO:common.agent.agents.abstract_gateway_agent:[covidphysio_telegram_gateway_agent] Response
↪ received.\n"}
{"log":"INFO:common.agent.agents.my_abstract_agent:[covidphysio_telegram_gateway_agent] Agent
↪ john-5fcdea5ab08efe960ba18f28@prosody.localhost asked for subscription. Let's approve it.\n"}
{"log":"INFO:covidphysio.telegram.agent.strategies.chat_platform.telegram_handling_strategies: Bound user
↪ ID `5fcdea5ab08efe960ba18f28` to Telegram ID `111111111`\n"}
{"log":"INFO:common.agent.agents.abstract_user_agent: Updated user last interaction, with the system, to:
↪ 2020-12-07 09:39:55.310185\n"}
{"log":"INFO:common.agent.behaviour.abstract_user_agent_behaviours:[john-5fcdea5ab08efe960ba18f28] Will
↪ handle `/start` from `telegram` with ID `1725`\n"}
```

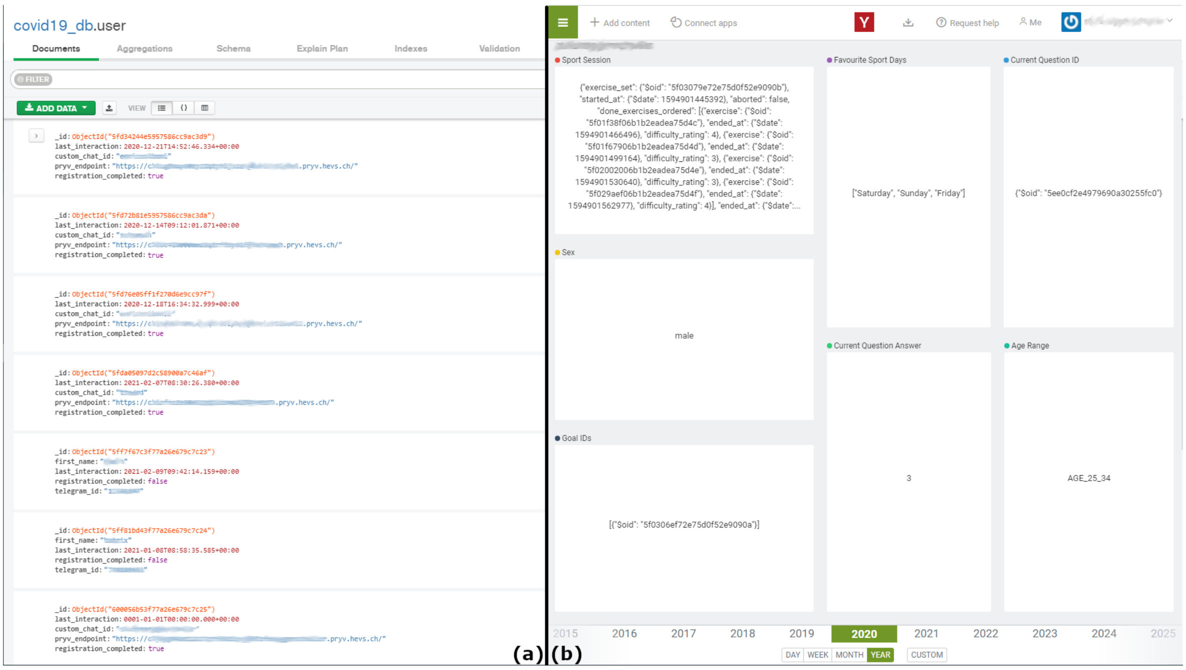


Figure 4. User objects in MongoDB (a) and User object in Pryv (b).

Listing 5 shows the method used by the Gateway Agent to forward the received messages to the respective user agent(s). The Gateway Agent is the connecting point between HemerApp/Telegram and the MAS. Thus, the message needs to *(i)* be converted into a Spade-conform format (i.e., a flattened and stringified dictionary representing the object – Line 4), and then *(ii)* a new MAS message object instance is created (Lines 6-11) and sent to the respective user agent by the Spade framework in the form of an XMPP message (Line 12).

Listing 5. Gateway Agent: Forward message to user agent.

```
1  def forward_chat_message(message: Union[types.Message, types.CallbackQuery], to_agent: Agent):
2      """Forwards received commands to provided Agent"""
3
4      preprocessed_telegram_message = preprocess_and_label_telegram_message(message)
5
6      mas_message = Message(
7          to=str(to_agent.jid),
8          sender=TELEGRAM_SENDER_NAME,
9          body=demojize(message.text) if isinstance(message, types.Message) else message.data,
10         metadata=preprocessed_telegram_message.strings_dictionary
11     )
12     to_agent.dispatch(mas_message)
```

Representative for all insertion states Listing 6 shows how the user agent handles the case of a missing language selection. For Telegram users, the interaction language is set according to the one specified in the app. If such a language is not supported by EREBOTS (i.e., English, French, Italian, and German), English is set as the default interaction language. For HemerApp users, a custom menu composed of four buttons (one per language) is directly presented to the user before any other action possible action. When executing the static method (lines 11-24), a message is sent to the user via the respective chat platform (Telegram or HemerApp). The message consists of a localized text (English by default due to lack of language selection) and a custom keyboard displaying the available language options to the user. These options are stored in an enumerator and defined on line 4. If the user now makes a valid selection using the custom keyboard, a message is sent to the selected front-end and traverses through the gateway to the user agent. The user agent then executes the function *on_legal_value* (lines 6). The selected language is extracted from the message and persisted in the user object (Lines 7-8) before a transition to the next state is performed (Line 9).

Listing 6. User Registration: LanguageInsertionState.

```
1  class LanguageInsertionState(AbstractDataInsertionState):
2      """A FSM State to handle user Language insertion"""
3
4      KEYBOARD_OPTIONS_NOT_LOCALIZED = [...] # i.e., "English", "French", "Italian", and "German"
5      ...
6      async def on_legal_value(self, user: AbstractUser, legal_value: str, sender_id: str):
7          uglyfier_dictionary = inverse_dictionary(Language.values_prettifier_dictionary())
8          user.language = Language(uglyfier_dictionary[legal_value])
9          await self.proceed_to_next_state(user, sender_id)
10
11     @staticmethod
12     async def ask_for_language(current_state: AbstractCovid19ReceiveMessageState, recipient_id: str,
13                                current_language: Optional[Language]) -> ChatMessage:
14         """Utility method to ask for Language"""
15
16         current_state.set_next_state(LanguageInsertionState)
17         return await messaging_platform.send_message_after_sleep(
18             recipient_id,
19             localize(LANGUAGE_QUESTION_TEXT_NOT_LOCALIZED, current_language),
20             custom_keyboard_obj=custom_keyboard_obj,
21             current_state.messaging_platform_handling_strategies.create_menu_keyboard_from(
22                 localize_list(LanguageInsertionState.KEYBOARD_OPTIONS_NOT_LOCALIZED, current_language)
23             )
24         )
```

As a best practice, each agent has at least one cyclic behavior used to parse incoming messages and react accordingly (see Listing 7).

Listing 7. User Agent: Receive MAS message behavior.

```python
class AbstractWaitForMessageState(State, ABC):
    """This is the main state in which we wait for the next message arrival"""

    async def run(self):
        msg = await self.receive(timeout=sys.maxsize)
        if msg:
            try:
                await self.on_message_received(msg)
            except:
                log_exception(self.agent)

        if self.should_set_next_state:
            super().set_next_state(self.STATE_NAME)
        else:
            self.should_set_next_state = True

    @abstractmethod
    async def on_message_received(self, mas_message: Message):
        """Template method called upon MAS message receiving, to handle it"""
        pass
```

3.1. Scenario, Functionalities, Dynamics, and Behaviors

This section describes EREBOTS's main functionalities, dynamics, and workflow. The developed platform has been tested and/or prototyped in the following scenarios:

- SC1 *Preventive physical conditioning:* it profiles the user according to a basic motor-balance assessment & his/her preferences and provides tailored exercises according to the user experience/profile both reactively and proactively.
- SC2 *Smoking cessations*: it consists of a 2-phases campaign. In phase 1, the bot determines the severity of the addiction (i.e., daily consumption, nicotine dependency) while recording the user's smoking habits. In phase 2, the bot assists the user during the craving episodes providing personalized mood boosters, health tips, behavioral tracking, feedback/reporting support, and adherence/efficacy evaluation.
- SC3 *Brest cancer survivors:* The bot provides informational content and advice according to the type of cancer, demographics, stage, physical condition, etc. The bot may counsel exercise sets targeting regaining/maintaining muscular strength and minimum physical activity levels.

3.1.1. Scenario SC1

In this section, we provide a more in-depth description of the functionalities, behavior, and tests related to scenario SC1, as it was developed in much more detail than the others. In particular, SC1 has been deployed in the context of the COVID-19 sanitary restrictions in Switzerland. Through its different stages, the *lockdown* involved social isolation, which, in many cases, consisted of a strict confinement. This situation implied restrictions to mobility and augmentation of sedentary habits, which may lead to a degeneration of motor functions (e.g., balance and strength) [66]. To counter this problem, we have collaborated with healthcare specialists in physiotherapy and rehabilitation at the Institute of Health at HES-SO Valais-Wallis to realize a chatbot assisting the user with personalized exercises. The physical therapy experts identified specific aspects to improve during the coaching program, such as balance or strength. For instance, regarding balance, they devised into 11 categories with 4 level of difficulties each. In the first stage, the user had to undertake a self-assessment consisting of a series of questions (see Table 1) whose outcome would define the difficulty level of the exercises to be proposed. On a scale from 1 to 5, where the latter is defined as *impossible*, the user is associated with a given class depending on this assessment. This categorization can be created and customized by the physical therapists through a web interface dedicated to the configuration of storylines for a given scenario.

3.1.2. Functionalities

Once the story line is created the system offers the following user functionalities (UF) and doctor functionalities (DF):

DF1: Create, modify, and delete objectives, exercises, and relationships among them.

Table 1. Set of questions for user balance self-evaluation

#	Question
1	How difficult is it for you to keep your balance when you stand in a quiet environment?
2	How difficult is it for you to keep your balance when you walk around in the apartment?
3	How difficult is it for you to keep your balance when you climb up a stair?
4	How difficult is it for you to keep your balance when you reach for an object that is on the table far in front of you?
5	How difficult is it for you to keep your balance when you pick something up off the ground?
6	How difficult is it for you to keep your balance when you stand on tiptoe to get a cup from the cupboard?
7	How difficult is it for you to keep your balance when you are being pushed by your pet or by someone or when you stumble over something?
8	How difficult is it for you to keep your balance when you carry a package to the apartment?
9	How difficult is it for you to keep your balance when you step down a stair?
10	How difficult is it for you to keep your balance when you walk and look back?
11	How difficult is it for you to keep your balance when you walk across the wet bathroom floor?

DF2: Visualize a single user and her aggregated information.
UF1: Register a new profile.
UF2: Manage his/her profile and settings (i.e., language[8], user goals, ability re-evaluation).
UF3: Ask for exercises (matching the user's level).
UF4: Visualize personal statistics and performance.
UF5: Get detailed information about the system functionalities and data usage, visibility, and storage.

Thanks to **DF1**, the physical therapist and/or healthcare personnel can define and customize several aspects of the campaign at run-time via the dedicated web application. In particular, the system allows to:

(i) Define the user goals, such as the desired level of balance to be attained.
(ii) Define the self-assessment questions, i.e., the set of questions to be asked to the user to determine her current situation with respect to the desired goals.
(iii) Associate the questions to a specific difficulty level.
(iv) Relate the questions to each other, defining the overall physical activity plan.
(v) Define the exercises to be suggested, including their instructions, and related multimedia (see Figure 5).
(vi) Assign the exercises to each difficulty level.

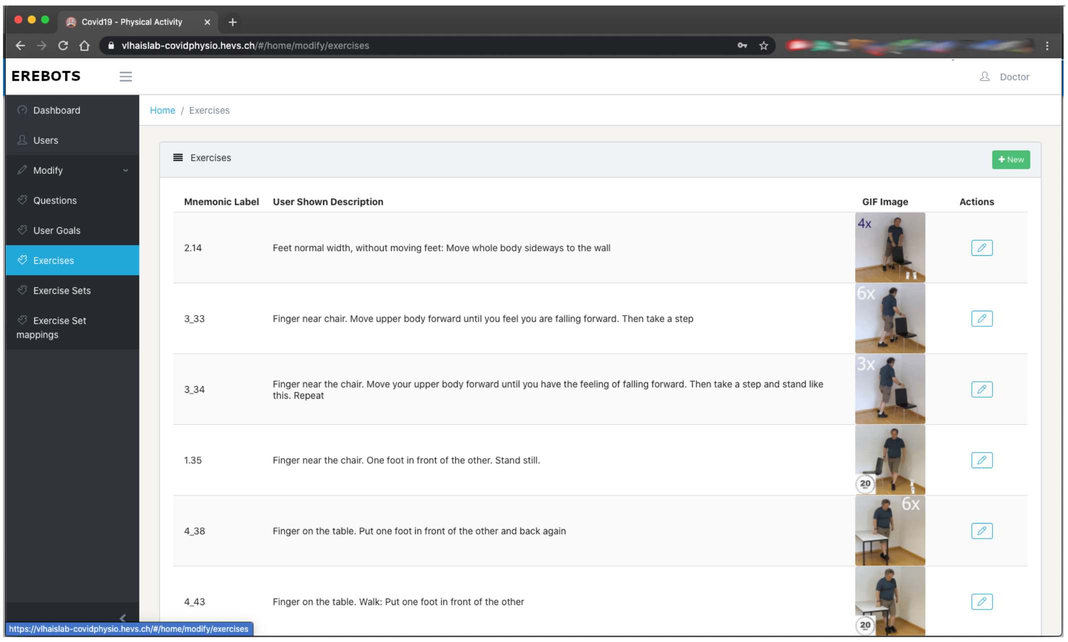


Figure 5. Schedule of the exercise sequence and upload of the related multimedia file.

[8] As of today, SC1 supports English, Italian, French, and German.

Concerning **DF2**, the physiotherapists and healthcare personnel are able to have a complete overview of the campaign and the general progress of the participants. More specifically, they have access to statistics, population composition in terms of gender, age group, language, physical advancement, etc. Figure 6 shows the dashboard visualizing synthetic data of a campaign managed by EREBOTS.

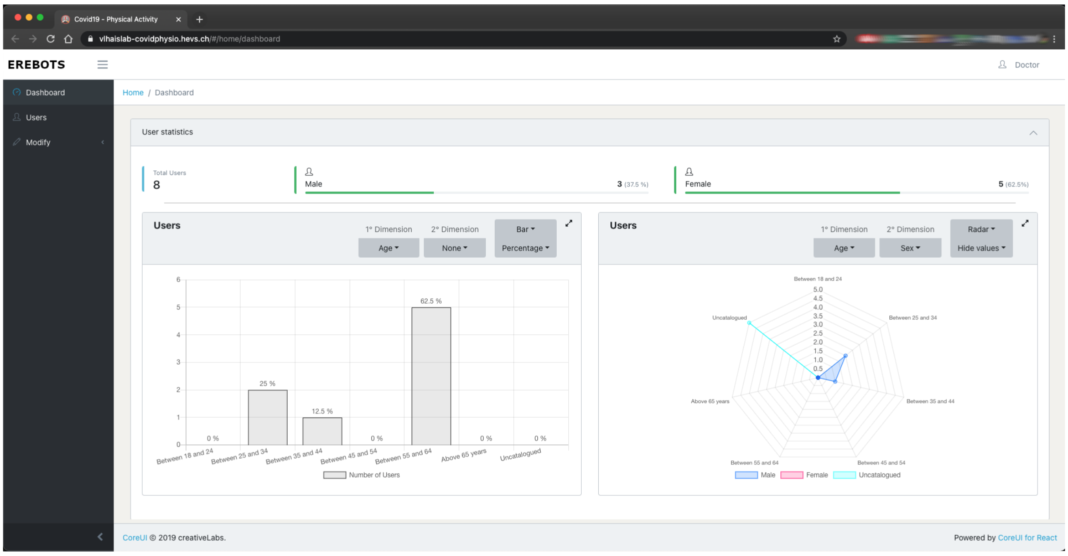


Figure 6. EREBOTS dashboard for healthcare personnel displaying the campaign in multi-dimensional graphs.

Concerning **UF1**, at the first access, the user is required to register a profile on Pryv.io and grant access to the specified information (see Figure 8). In this way, the user has control over which information is shared with the EREBOTS framework in a fine-grained manner. Concluded the registration, the system generates a unique token that is used to associate the user to his/her personalized virtual agent. Figure 7 shows an interaction diagram characterizing the login process (from either Telegram or HemerApp).

When a user sends a message to the chatbot for the first time (regardless of the interface), the login process is triggered. The login process is roughly divided into three steps. First, the user is informed that all sensitive data is stored on Pryv, and therefore, a Pryv account is mandatory. If the user agrees to these terms, the DoctorAgent requests a unique authentication URL from the Pryv.io backend and forwards this to the user.

In a second step, the user logs in via the URL using their Pryv credentials (see Figure 8a), at which point a consent window is displayed that lists which permissions and data the chatbot would like to read and write (see Figure 8b). Once the user has accepted the consent form and notifies the chatbot, the DoctorAgent performs the final step to obtain the user's authentication code by polling the Pryv.io back-end.

To allow basic user-personalization, the chatbot asks the user for additional personal information such as language, name, age, sex, favourite days for sport, physical goals (see Figures 9a, 9b, and 9c). The initial procedure concludes with user self-assessment of his/her basic physical abilities (see Figure 10a) functional to the purpose of the given campaign (see Table 1). In turn, the user can freely interact with the chatbot and explore the functionalities of HemerApp (see Figure 10b).

The user can tap on the "update profile" button, receive the summary of his/her profile, and update it at any time, fulfilling **UF2**. Regarding **UF3**, the user can request at any time a set of exercises tailored for his/her level. The bot proposes one or more sets to the user who can decide whether to change it, start, or go back (Figure 10c). When the user starts, a popup is triggered displaying the instructions and multimedia that describe how to do the exercise and the commands to *start, pause, restart, complete,* and *abort* the exercise (Figure 11a). Once each exercise is completed, the chatbot asks for a self-evaluation (Figure 11b). At the completion of each exercises session, the chatbot provides a summary with *exercise, the time elapsed*, and *difficulty feedback*. To better tailor the exercise distribution and understand the user acceptance, the bot asks to rate the session (Figure 11c).

As for **UF4**, the user can visualize the overall use of the application in terms of user/chatbot/total *messages exchanged*, completed/interrupted/total *training sessions,*, and *training time* (Figure 12a). Moreover, to track the evolution of the user, the system proposes an interactive graph (i.e., taping on each point provides further details)

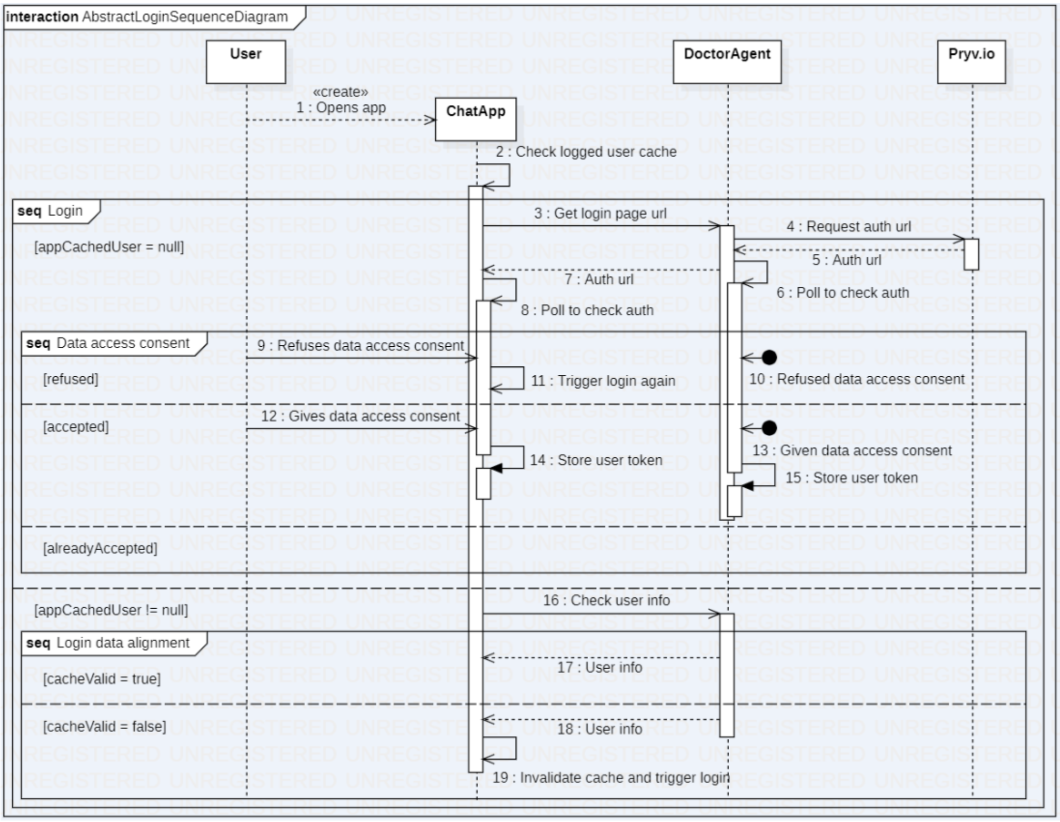
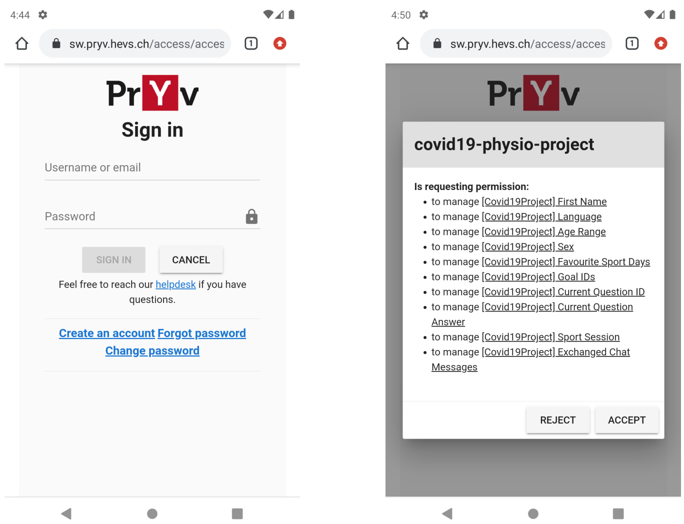


Figure 7. Front-end login interaction diagram via Telegram or HemerApp, and data-flow between User, ChatApp, doctor-agent, and Pryv.

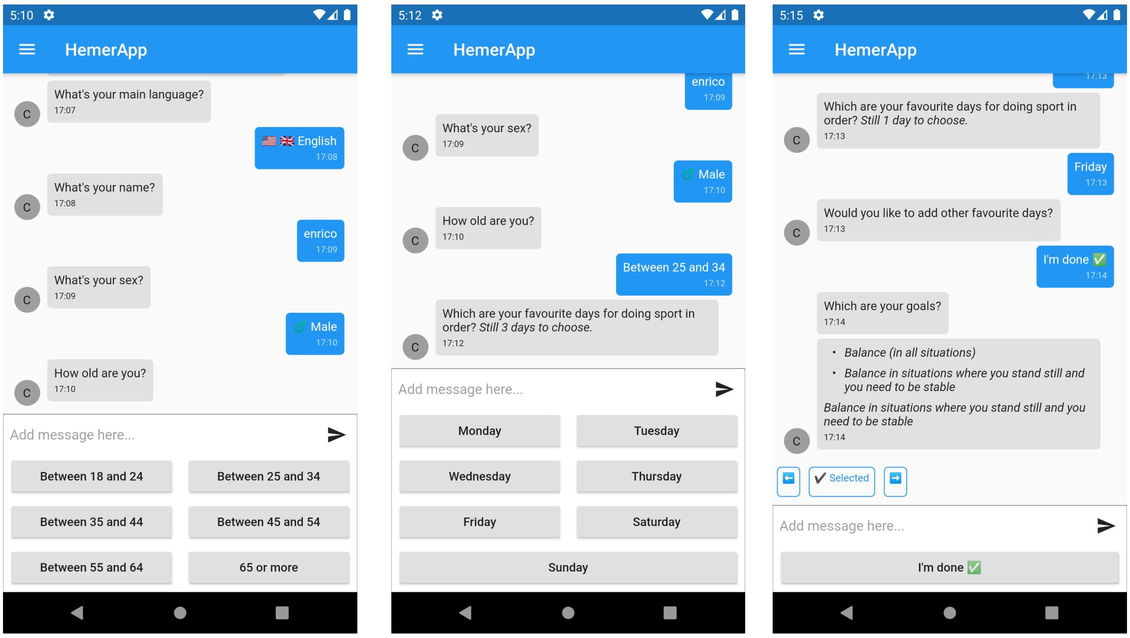


(a) Pryv interface to login and register.

(b) Description of the privacy grant for Pryv access.

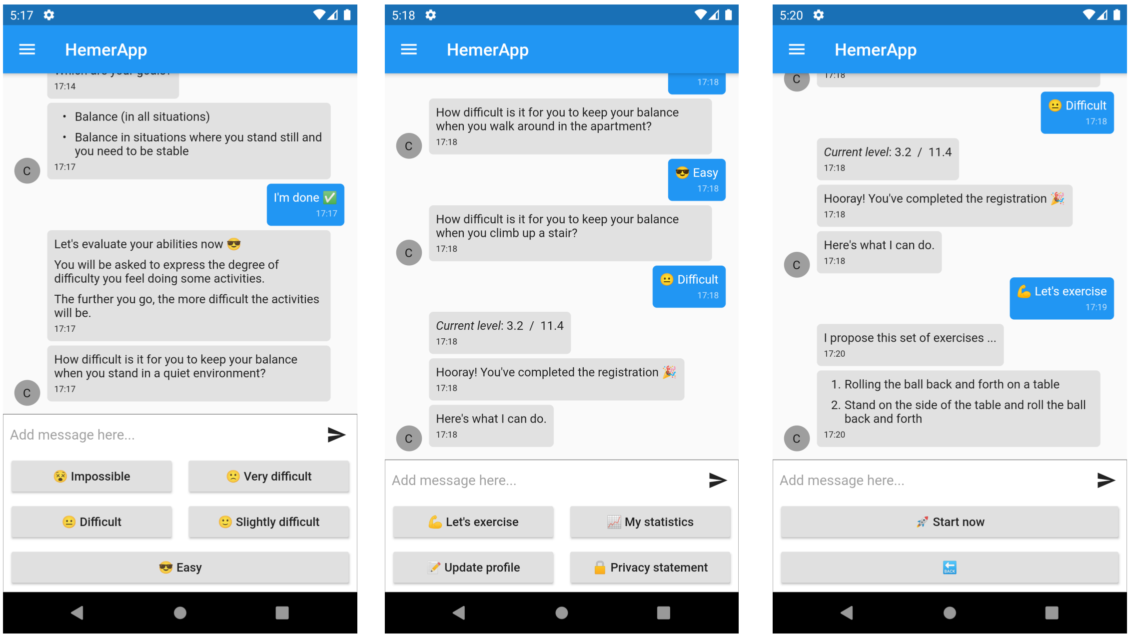
Figure 8. Interfaces for registration, login, and granting access to personal information.

concerning the training trend with respect to the difficulty level (Figure 12b). In addition, and following **UF5**, the app provides a view of the information about the changelog of the application interface. Finally, Figure 12c shows a dynamically generated data privacy statement. As opposed to other systems where this statement is static (usually written by the developers), a dedicated behavior inspects all the system's functionalities/behaviors handling data, and provides a report that is displayed to the user. In such a way, fostering transparency, human mistakes or information omission can be avoided.



(a) Language & age selection. **(b)** Physical activity day preferences. **(c)** Goal(s) selection.

Figure 9. Demographics, preferences and goals selection for user profiling in HemerApp.

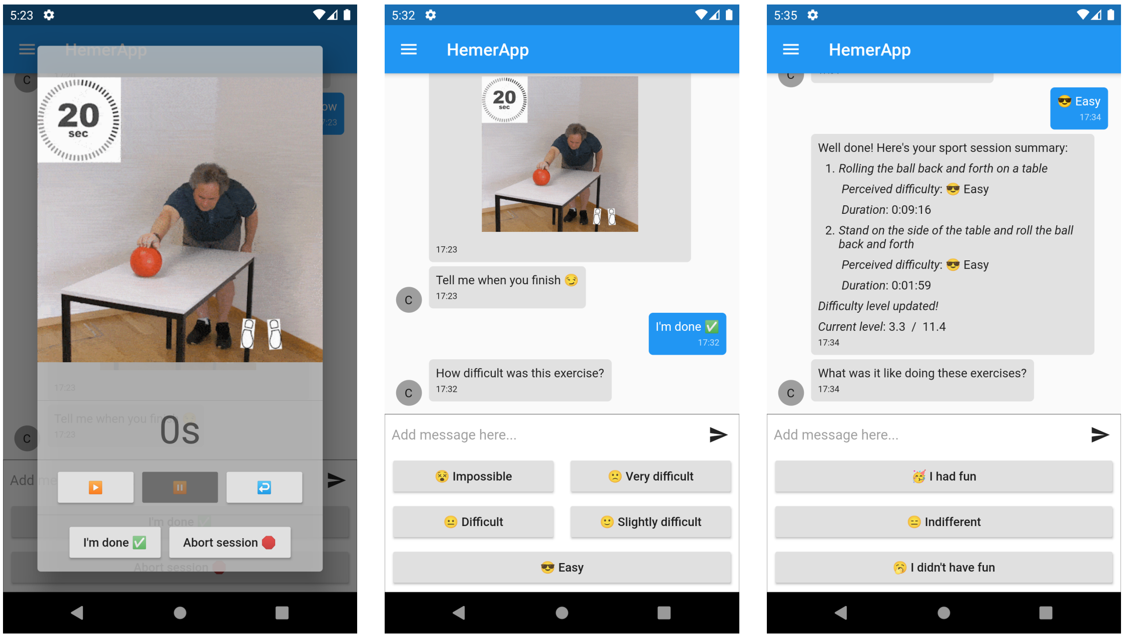


(a) Self-assessment of physio-motor conditions. **(b)** Main menu after profile setting and self-assessment. **(c)** Request for exercises tailored to the user level.

Figure 10. Self-assessment and tailored exercise request in HemerApp.

The behavior of the User Agent for the COVID-19 physical balance preservation can be schematized as shown in Figure 13.

The overall message-exchange characterizing the dynamics presented above is schematized in Figure 14. Notice that before any interaction, the ChatApp opens a connection to EREBOTS through the GatewayAgent. The GatewayAgent has two main roles. Firstly, it acts as a gateway for messages sent via the two interfaces (Telegram or HemerApp) to the chatbot, and if the HemerApp interface is used, it stored the open chat connections. Secondly, it manages the creation of UserAgents in case a new user contacts the bot. The DoctorAgent may send any message events depending on the current behavior status of the user. The ChatApp is ready to receive any

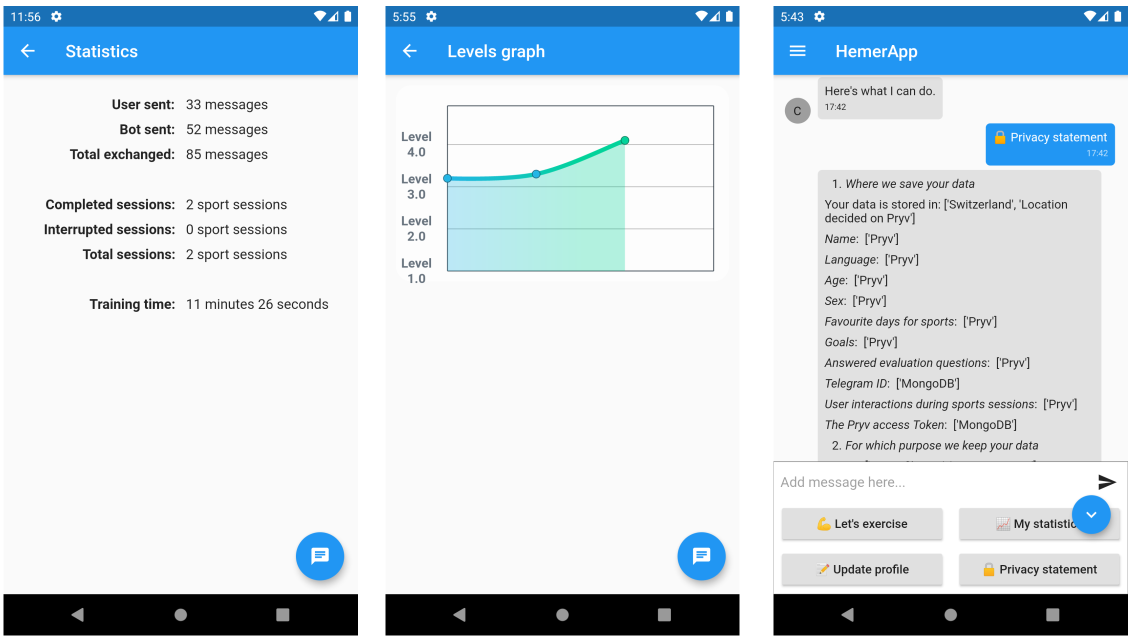


(a) Dedicated interface to control the exercises' execution.

(b) Rating the difficulty of the exercise just concluded.

(c) Summary of the exercises and feedback for the overall experience.

Figure 11. Exercise execution, evaluation and overall set appreciation in HemerApp.



(a) Usage Statistics.

(b) User level information.

(c) Dynamic data privacy statement.

Figure 12. User's usage statistic, training's difficulty trend, and data privacy statement.

input from the user, which may be redirected to the UserAgent for further processing. All chat messages are stored in the personal data store in Pryv.

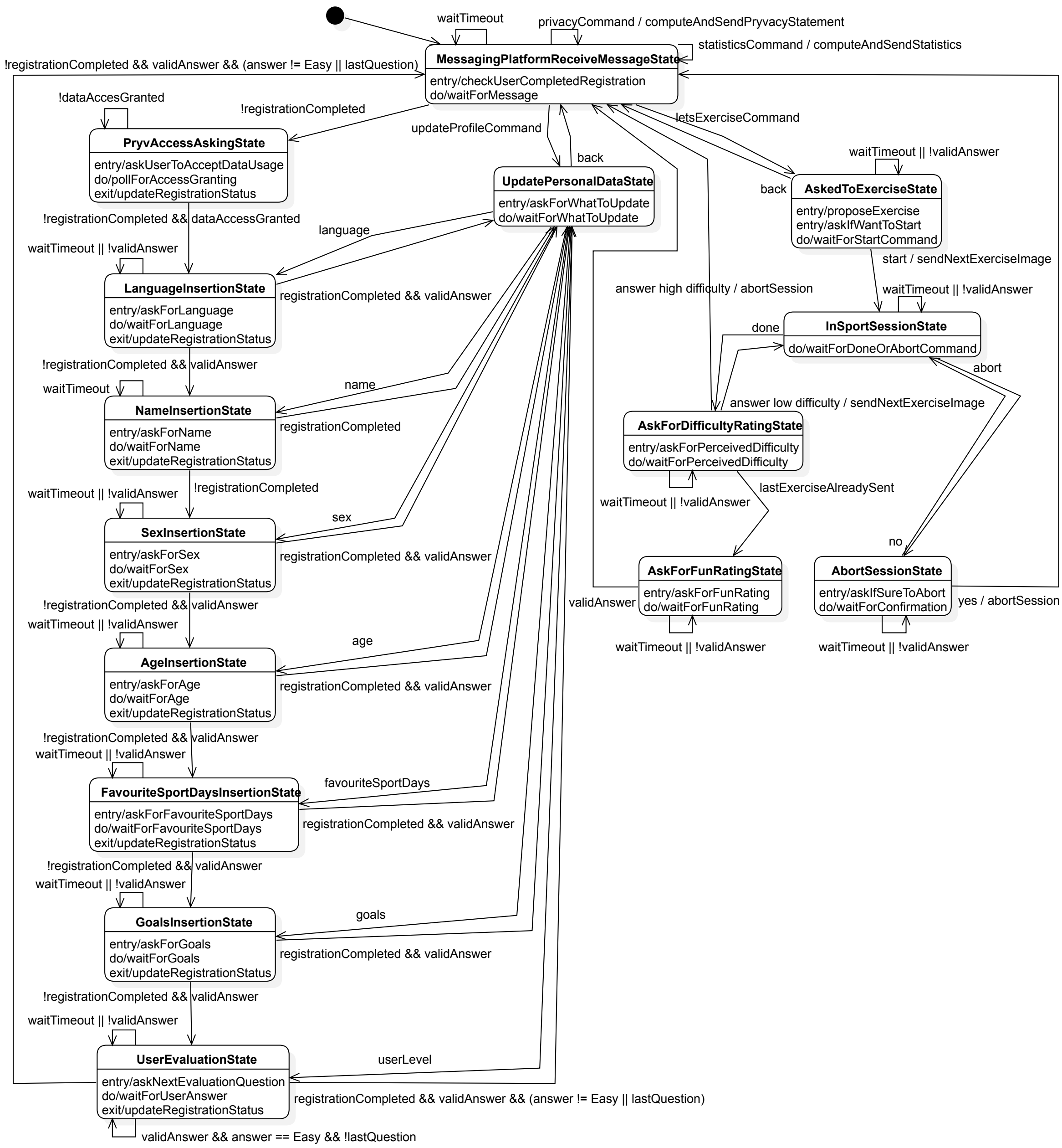


Figure 13. Schematic representation of the UserAgent behavior.

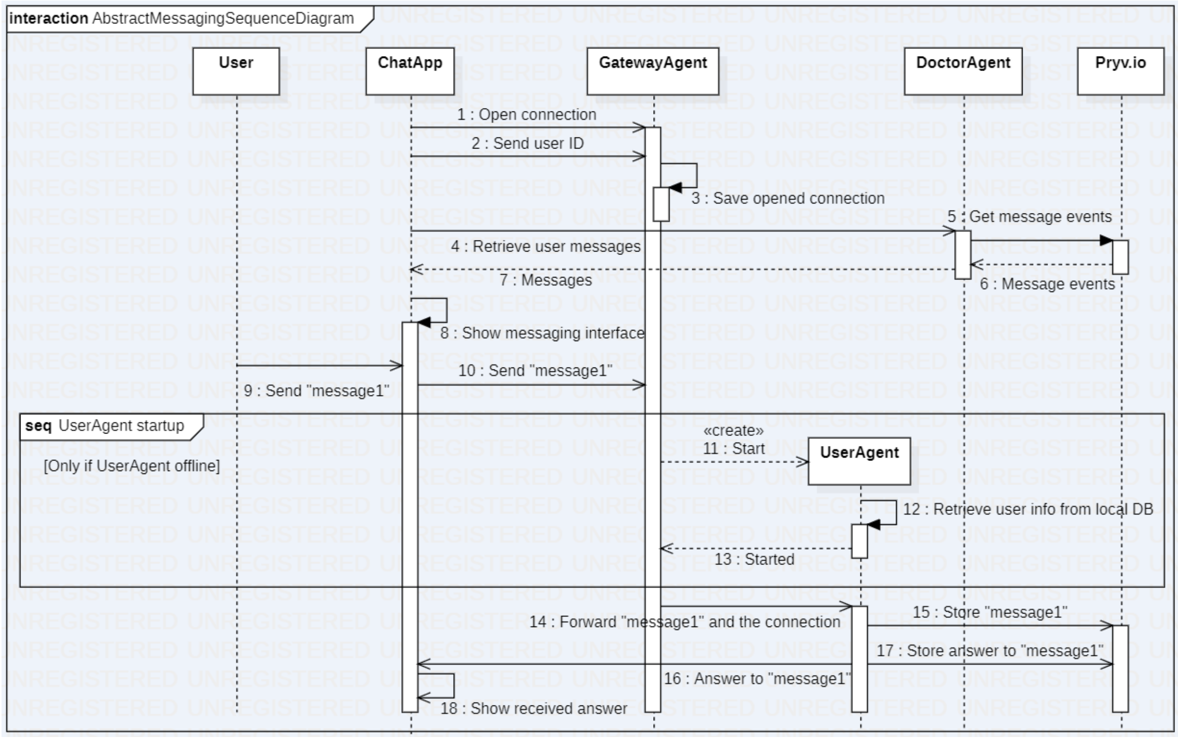


Figure 14. Messaging process interaction diagram.

4. Experimentation

To test EREBOTS and HemerApp, we involved 13 participants, hereafter referred to as u_x with x ranging from 1 to 13 for a total duration of 12 days in August 2020. Such a population is characterized by 7 women and 6 men, living in Switzerland (6), Italy (4), and France (3), whose selected in interaction language is English (3), French (3), Italian (4), and German (3). Moreover, testers are composed of individuals from 18 to 65+ years old equally distributed among six classes and recorded a difficulty entry-level as show in Table 2.

Table 2. Difficulty entry-level per individual.

Individual	Difficulty entry-level	Description
U_1	7	Difficult to keep balance being pushed.
U_2	4	Difficult to reach objects far on a table.
U_3	5	Difficult to pick something from the ground.
U_4	8	Difficult to keep the balance while carrying a *medium/big* package.
U_5	8	Difficult to keep the balance while carrying a *medium/big* package.
U_6	5	Difficult to pick something from the ground.
U_7	9	Difficult to keep balance stepping downstairs.
U_8	2	Difficult to constantly keep balance when walking.
U_9	7	Difficult to keep balance being pushed.
U_{10}	3	Difficult to keep balance when climbing stairs.
U_{11}	4	Difficult to reach objects far on a table.
U_{12}	10	Difficult to keep balance when walking while looking back.
U_{13}	10	Difficult to keep balance when walking while looking back.

Figure 15 shows the overall number of messages exchanged per user cluster. Among them, the users in two classes ($(45 - to - 54)$ and $(55 - to - 65)$) have shown a remarkably higher level of engagement, shown by both the total number of messages and the exercise sessions recorded.

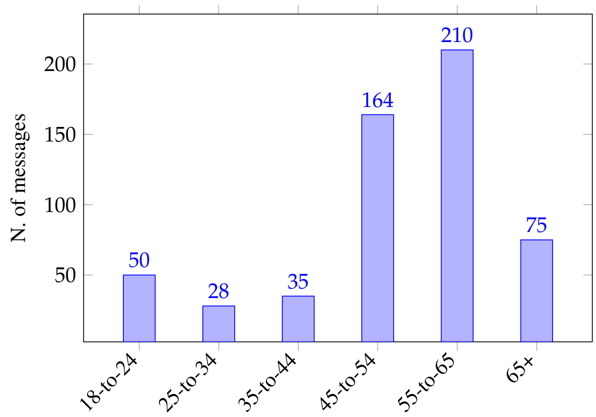


Figure 15. Average number of messages sent per participant within the age groups.

Figure 16 shows the overall number of messages per participant with a total mean of 87.76 of messages sent. From the figure, we see that u_2 sent the maximum number of messages (315 messages), whereas u_9 sent the minimum number of messages (14).

The number of messages sent is strongly related to the number of exercise sessions. Figure 17 illustrates the overall number of exercising sessions per participant. From the both Figures 16 and 17, we remark the positive correlation between the number of messages and the number of exercise sessions per participant. This correlation is function of the number of exercises present in each session, which involves diverse numbers of user-chatbot interactions. Indeed, although u_8 has fewer total messages than u_4, he/she has initialized more exercising sessions. This non-linear correlation is due to the users' answers to the chat-bot questions, which change the amount of information required by the bot. It is worth highlighting that user u_9 has not started any exercise session.

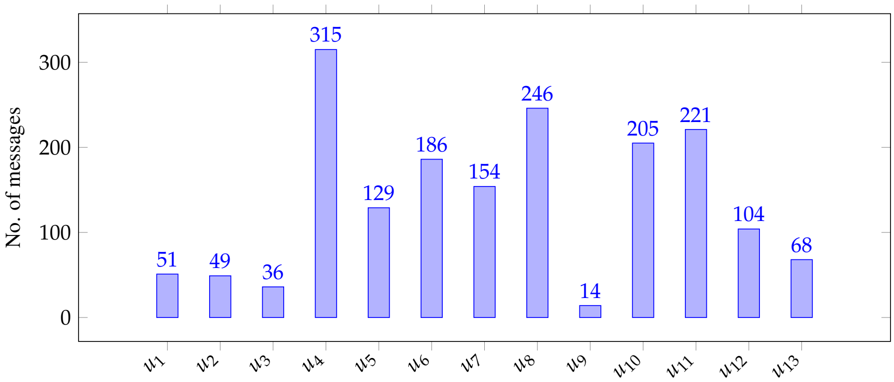


Figure 16. Number of messages per participant.

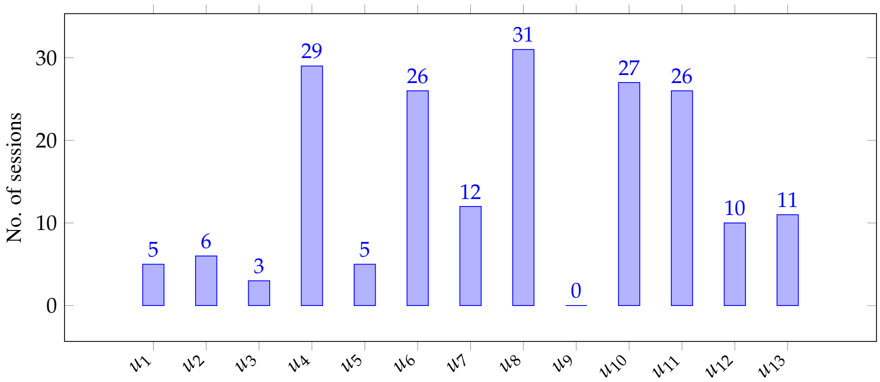


Figure 17. Number exercising sessions per participant.

Figure 18 shows the number of completed exercises per participant. From the figure, we notice that user u_4 (who has the maximum number of messages exchanged) has completed the most number of exercises (64). On the other hand, since the number of exercises varies per exercising session, u_8, which initialized the maximum of the exercising session, completed fewer exercises than u_4. This reasoning applies as well to users u_{10-13}.

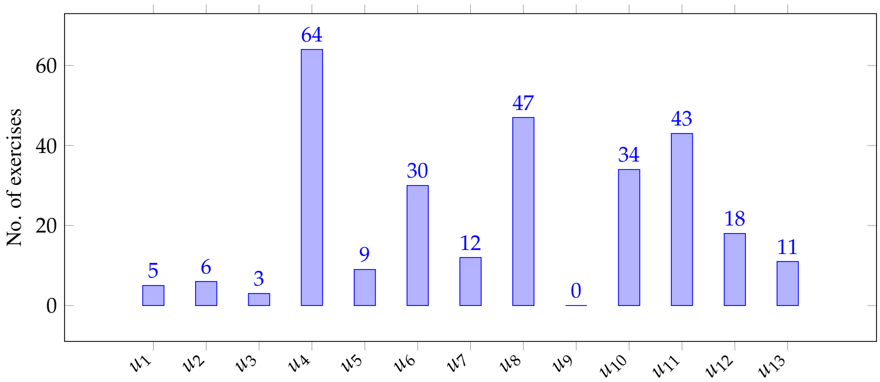


Figure 18. Number of completed exercises per participant.

Figure 19 shows the number of aborted exercises per participant. Overall, during the entire testing period, only (9) exercises have been aborted, which is less than the 5% of the total exercises initiated.

After the initial self-assessment (which can be re-executed at any time), the difficulty of the upcoming exercises proposed to the user is based on his/her previous evaluations/feedback. On the one hand, Figure 20 reports the advancements in terms of difficulty levels per participant. On the other hand, Figure 21 shows the regressions (only a total of 5 among 13 users). Such a situation suggests two possible reading keys: most of the users have initially underestimated their actual level, and/or the difficulty gap among the level is well-tuned and allows an effective gradual progression. However, this latter can be just a personal interpretation. Indeed, comparing Figures 18 and 20, it is possible to notice that user $u_1 0$ advanced more difficulty levels than u_4 (who

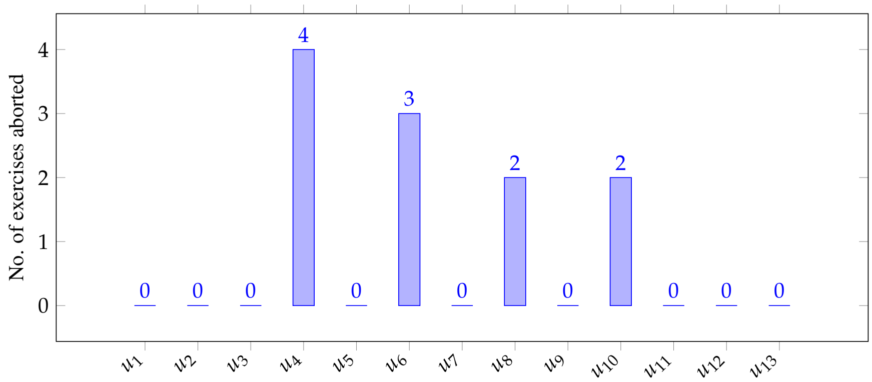


Figure 19. Number of aborted exercises per participant.

completed the most exercises). Such behavior is inducted by the personalized nature of the run-time exercises assignment, which, in this first version, is mainly coupled with the user difficulty perception. Such a feedback mechanism induces the system to quickly converge to a more appropriate difficulty level according to the user judgment.

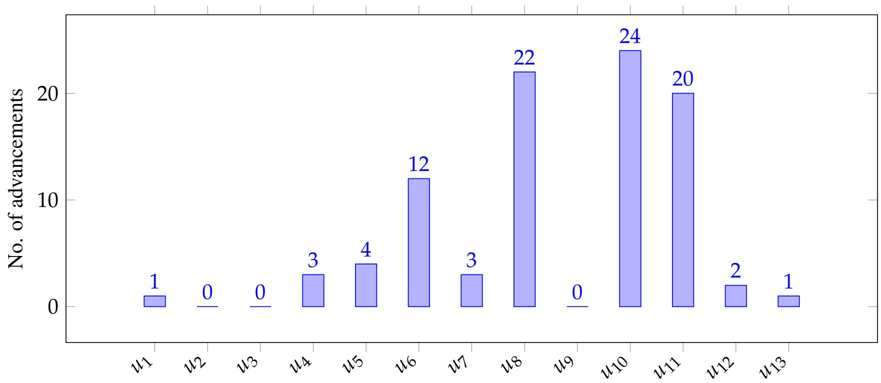


Figure 20. Number of advancements in terms of difficulty levels per participant.

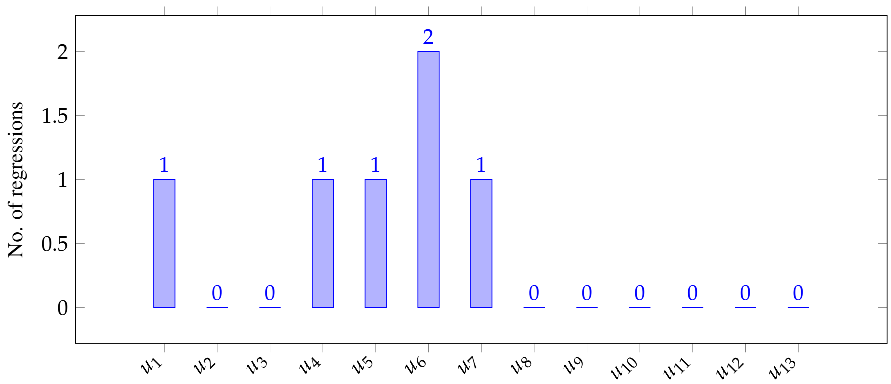


Figure 21. Number of regressions in terms of difficulty levels per participant.

Concerning the user satisfaction, the summary of all the evaluations provided by the users about each set of exercises is shown in Figure 22. Overall, it is possible to assert a majority of *positive* feedback (92) followed by *indifferent* (56), and only (17) *negative*. The negative/indifferent feedback have been used by the medical personnel supervising the test to understand better the user-exercise coupling and advance in the formulation of a personalized user model.

In terms of system performance, the messages' response time (time elapsed from the moment a user has sent a message to the moment he/she receives a reply) recorded during the testing is shown in Figure 23. Overall, the mean is centered on 2 seconds, which defines an optimal trade-off in terms of human usability. Nevertheless, a few outliers have been recorded. Such specific situations have been generated by the users who carried out the

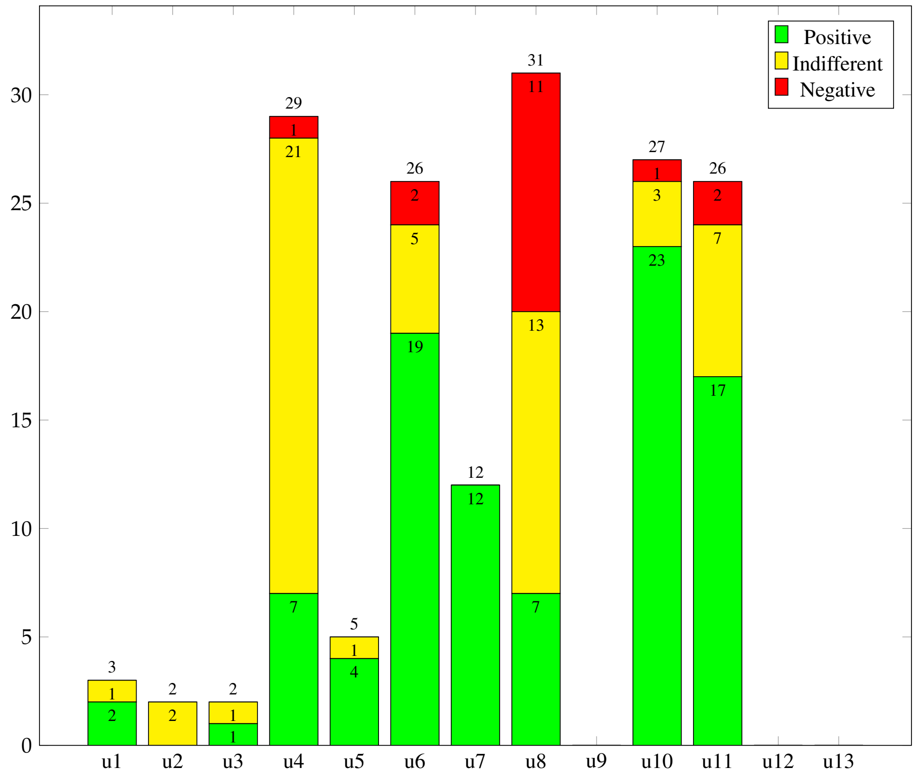


Figure 22. Users' feedback on Exercises sets

testing over Telegram and triggered a *security time-out* imposed by the platform to prevent flooding risks[9]. In HemerApp, such limitations are not necessary since the chatbot's behavior is ruled by in-house developed agents.

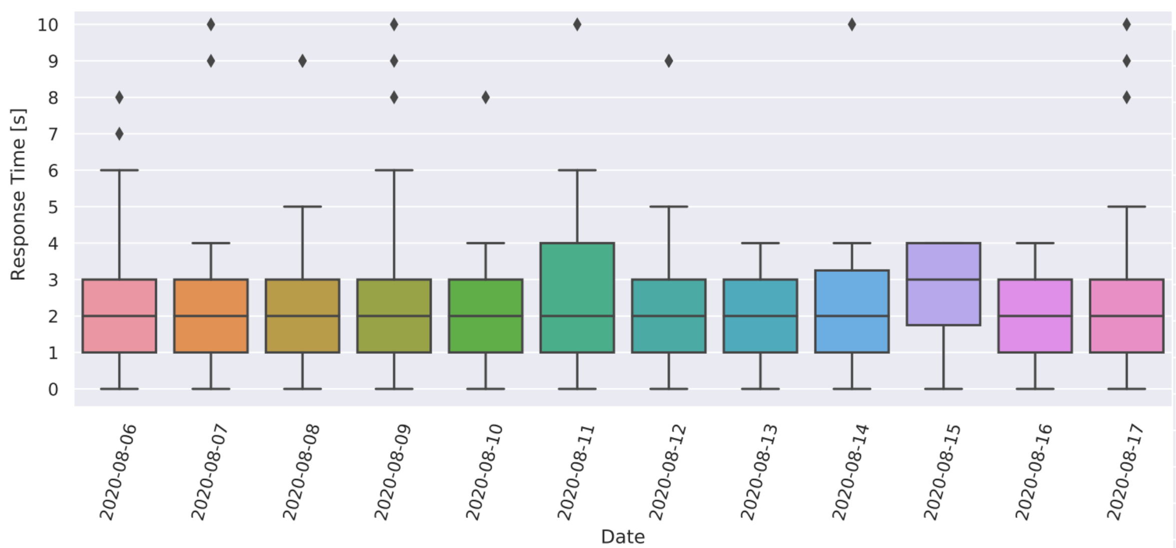


Figure 23. Messages response time over testing period.

Figure 24 provides a comprehensive overview of the users' behaviors during the testing phase. In particular, it is possible to see the time (hour/day) of any exchanged message per user and the related sum during the day

[9] In Telegram, any third-party can use the chatbot APIs. Therefore, to limit the chatbot traffic, Telegram has applied limits for the interleaving of messages containing multimedia files or being *heavier* than a given limit – https://github.com/python-telegram-bot/python-telegram-bot/wiki/Avoiding-flood-limits

and day out of the entire period. We can notice that most of the interactions have crowded between 7:00-10:00 and between 12:00-14:30. In terms of involvements over the days, most of the interactions occurred in the fourth day, followed by a gradual relapse to then increase again.

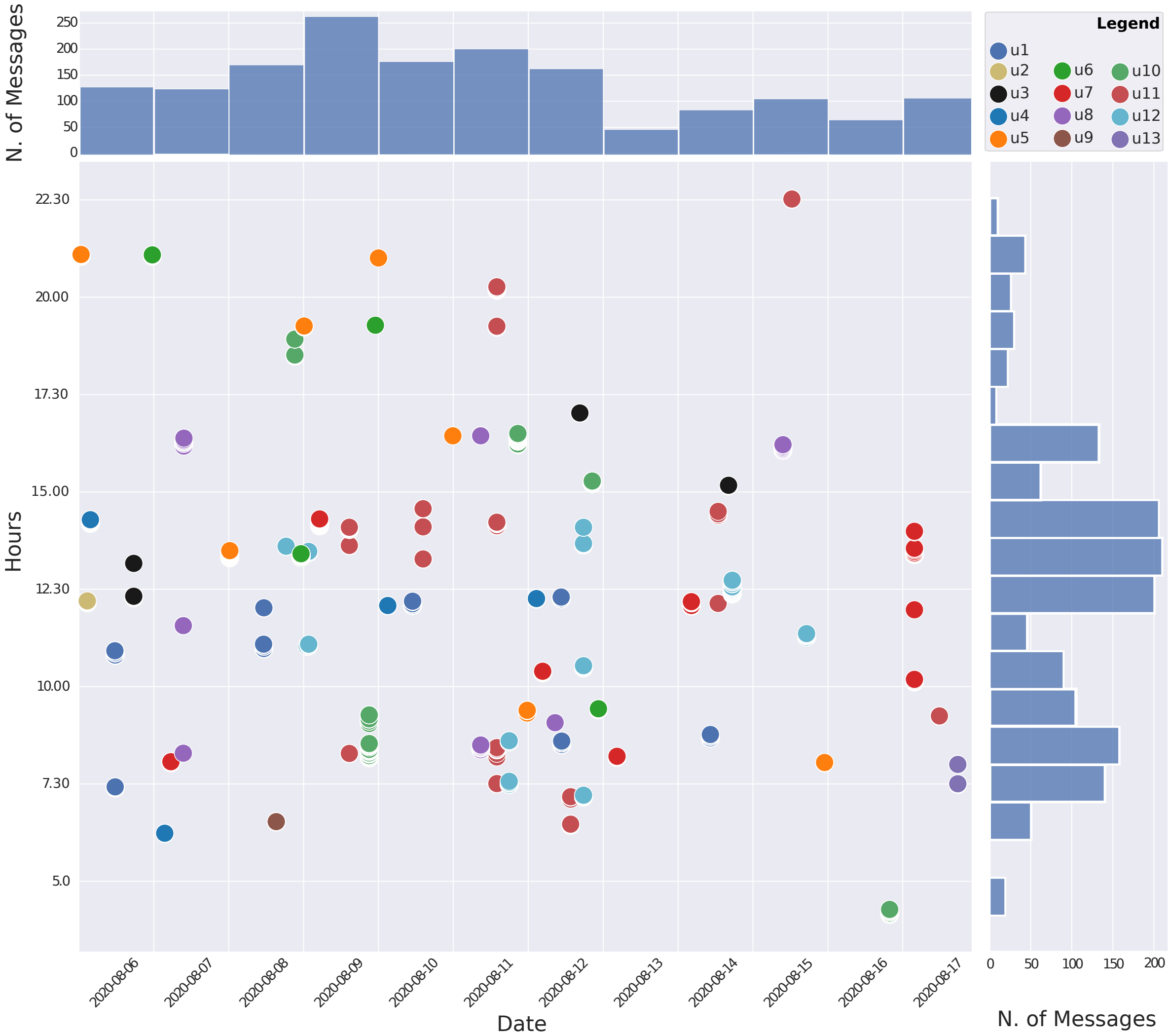


Figure 24. Overall users interaction with EREBOTS over the testing period.

5. Discussion

The design and implementation principles of EREBOTS and its mobile interface HemerApp have been inspired by the features and challenges described previously in Section 2.5. Next, we discuss how the framework and the results address these challenges and to what extent.

First, regarding the ability to implement social interactions among agents **C1** (Social A2A), it is worth recalling that each human user is embodied by a virtual agent. This has made it possible for agents to engage in back-end interactions (A2A), which may include sharing knowledge and autonomously pursuing both personal and common related goals (i.e., campaign) via FIPA-compliant message exchange. While the A2A approaches ensure clear advantages relying on the inherited benefits of the agent-based approach, the investigation of possible synergies between EREBOTS and non agent-based frameworks remain to be explored, with particular emphasis on strategies to automatize the knowledge exploration.

For the specific case of doctor (or healthcare provider) agents, EREBOTS provides an initial set of tools to monitor in real-time the running campaign. Such features partially address **C2** (run-time healthcare supervision). Indeed, we are working to satisfy this challenge fully, and we plan to extend our mechanisms with logic-based triggers to involve proactively medical personnel when needed. Moreover, we will deploy specific mechanisms to enable medical specialists to take over the conversation from the bot.

Concerning modeling **C3** (evolving models & behaviors), the user modeling and knowledge representation can be dynamically reshaped to satisfy possibly different investigations/campaigns. As of today, the parallel execution of multi-campaigns is possible. Yet, besides possible, the seamless integration of contextually diverse knowledge is an ongoing work.

Besides multi-campaign capabilities, the challenge of multi-stakeholder personalization **C4** is considered in EREBOTS, specifically through fine-tuned data- and action-driven penalization. Moreover, the user-agents can be associated with specific classes (e.g., roles) and receive personalized mainstream interaction story-lines. Nevertheless, we understand that medical personnel might need functionalities that go beyond the in-chat personalization/differentiation. Therefore, as ongoing work, we are analyzing how to dynamically integrate user-groups dedicated to enriching the chatbot interface (HemerApp) and its interactions. Indeed, as often stated by the current state of art, not all the functionalities can reasonably occur in a text/menu-based chat.

In terms of Quality of Experience **C5** (users' QoE), the web interface and specific agent behaviors are in charge of punctually collecting users' feedback related to the tasks conducted within the application (e.g., exercise feedback). Nevertheless, although deeply related to the potential engagement that the user may have throughout the campaign, this is actually part of the process of personalization (as explained above). As ongoing work, we are studying the automation of such a feedback classification and placing autonomous logic triggers for sensitive feedback requiring the attention of the personnel managing a given campaign.

This dynamicity in the implemented agent behaviors **C6** is at least partially present in EREBOTS. While the backbone functionalities are standard (agent generation, security token registration, etc.), it is possible to (re)define at run-time several interaction patterns. For example, in SC1, the medical personnel has full control in composing and connecting stages and dynamics of the given story-line. As ongoing work, we are investigating the extent to which it is reasonable to allow the run-time definition of actual agents' behaviors. While it can be a remarkable advancement for the platform, it might introduce unwanted side effects.

Concerning **C7** (semantics & terminology), the system currently relies on semi-structured message exchange among agents. The data schema is defined as Pryv streams typically serialized in JSON. Although Pryv has the ability to expose its data using semantically rich representations [12] and to use standard vocabularies (e.g., HL7 FHIR), these still need to be incorporated into the EREBOTS implementation.

Regarding **C8** (delegation), the entanglement user - chatbot - personnel supervising the campaign might go beyond the simple automation of possibly machine-delegable behaviors. EREBOTS provides (pro)active mechanisms that have been tailored to the specific case study. Nevertheless, the generalization of such an assessment and the definition of proper boundaries still remains an open challenge.

Finally, concerning **C9** (privacy compliance). EREBOTS employs Pryv as a privacy-compliant stream-based database. Moreover, when the platform is deployed, an automated behavior composes an informative scrutinizing all the agents' behaviors within the system and collects *which data* is used for *which purpose* and visible to *who*. If a new behavior is added into EREBOTS or an existing one is modified, the informative is entirely recomposed.

6. Conclusions

In the context of personalized chatbots as virtual assistants, this paper coped with challenges such as agent-to-agent interaction, continuous healthcare personnel supervision, evolving models and behaviors, multi-stakeholder personalized therapy & persuasion, continuous QoE monitoring, dynamic mechanisms update, semantics & terminology, task delegation, and privacy compliance.

To this end, it presented an agent-based framework named EREBOTS and its related user interface named HemerApp to realize chatbots with multi-front-end connectors and interfaces (i.e., Telegram, dedicated App & web interface). Moreover the framework allows to implement and run parallel multi-scenarios behaviors, deploy personalized conversations and recommendations, and provide a responsive multi-device monitoring interface.

Such a platform has been tested in a physical exercise support scenario in the context of social-confinement situations, which allowed us to discuss the extent of satisfaction of the above-mentioned challenges. Overall, we have shown that *(i)* assistive agents can interact with each other in the back-end, opening the door to knowledge sharing for campaign-related investigations, *(ii)* medical personnel has access to real-time aggregated and personal information of the individuals participating in a given campaign, *(iii)* enabled multi-model knowledge representation can be enabled for simultaneous campaign executions, *(iv)* it is possible to fine-tune data-/action-driven personalization strategies, *(v)* user QoE can be monitored via direct feedback collection, *(vi)* it is possible to (re)define online therapies and campaigns story-lines, *(vii)* the data schema is defined as Pryv streams typically serialized in JSON and possibly exposed using semantically rich representations (e.g., HL7 FHIR – ongoing work in EREBOTS), *(viii)* (pro)active mechanisms can be tailored to a specific case study, and*(ix)* users' data are stored in a stream-based privacy-compliant system solely managed by the user.

Finally, it is worth highlighting that the testers have mostly provided positive feedback and recorded improvements w.r.t. their initial balance conditions.

Author Contributions: Conceptualization, D.C. and J.P.C.; resources, D.C. and J.P.C.; system design, D.C.; system development, E.S. and S.E.; system validation, D.C., S.E., and E.S.; test-bed scenario design and validation D.C. R.H. and J.P.C.; results analysis and interpretation, R.H., D.C., G.M., and S.E.; investigation and analysis, D.C. and J.P.C.; writing–original draft preparation, D.C., J.P.C., and G.M.; writing–review and editing, D.C., J.P.C., G.M., and M.S.; project supervision C.D., J.P.C., R.H., and M.S.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. López, G.; Quesada, L.; Guerrero, L.A. Alexa vs. Siri vs. Cortana vs. Google Assistant: a comparison of speech-based natural user interfaces. International Conference on Applied Human Factors and Ergonomics. Springer, 2017, pp. 241–250.
2. Johannsen, G. Human-machine interaction. *Control Systems, Robotics and Automation* **2009**, *21*, 132–62.
3. Weizenbaum, J. ELIZA—a computer program for the study of natural language communication between man and machine. *Communications of the ACM* **1966**, *9*, 36–45.
4. Calvaresi, D.; Calbimonte, J.P.; Dubosson, F.; Najjar, A.; Schumacher, M. Social Network Chatbots for Smoking Cessation: Agent and Multi-Agent Frameworks. 2019 IEEE/WIC/ACM International Conference on Web Intelligence (WI). IEEE, 2019, pp. 286–292.
5. Xu, A.; Liu, Z.; Guo, Y.; Sinha, V.; Akkiraju, R. A new chatbot for customer service on social media. Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems. ACM, 2017, pp. 3506–3510.
6. Calbimonte, J.P.; Calvaresi, D.; Dubosson, F.; Schumacher, M. Towards Profile and Domain Modelling in Agent-Based Applications for Behavior Change. International Conference on Practical Applications of Agents and Multi-Agent Systems. Springer, 2019, pp. 16–28.
7. Fadhil, A.; Gabrielli, S. Addressing challenges in promoting healthy lifestyles: the al-chatbot approach. Proceedings of the 11th EAI International Conference on Pervasive Computing Technologies for Healthcare. ACM, 2017, pp. 261–265.

8. Graham, A.L.; Papandonatos, G.D.; Erar, B.; Stanton, C.A. Use of an online smoking cessation community promotes abstinence: Results of propensity score weighting. *Health Psychology* **2015**, *34*, 1286.
9. Roca, S.; Sancho, J.; García, J.; Alesanco, Á. Microservice chatbot architecture for chronic patient support. *Journal of biomedical informatics* **2020**, *102*, 103305.
10. Ni, L.; Lu, C.; Liu, N.; Liu, J. Mandy: Towards a smart primary care chatbot application. International symposium on knowledge and systems sciences. Springer, 2017, pp. 38–52.
11. Voigt, P.; Von dem Bussche, A. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing* **2017**.
12. Calbimonte, J.P.; Dubosson, F.; Kebets, I.; Legris, P.M.; Schumacher, M.I. Semi-automatic Semantic Enrichment of Personal Data Streams. SEMANTICS Posters&Demos, 2019.
13. AALex. https://aws.amazon.com/lex/. Accessed: 2018-09-29.
14. DialogFlow. https://dialogflow.com, 2018. Accessed: 2020-12-29.
15. Microsoft. Microsoft Bot Framework. https://dev.botframework.com. Accessed: 2020-12-29.
16. SAP. SAP Conversational AI. https://www.sap.com/products/conversational-ai.html, 2020. Accessed: 2021-01-20.
17. Rasa. Rasa Open Source. https://rasa.com/, 2020. Accessed: 2021-01-20.
18. Chaves, A.P.; Gerosa, M.A. How Should My Chatbot Interact? A Survey on Social Characteristics in Human–Chatbot Interaction Design. *International Journal of Human–Computer Interaction* **2020**, pp. 1–30.
19. Lee, M.; Lucas, G.; Mell, J.; Johnson, E.; Gratch, J. What's on Your Virtual Mind? Mind Perception in Human-Agent Negotiations. Proceedings of the 19th acm international conference on intelligent virtual agents, 2019, pp. 38–45.
20. Meany, M.M.; Clark, T. Humour Theory and Conversational Agents: An Application in the Development of Computer-based Agents. *International Journal of the Humanities* **2010**, *8*.
21. Maroengsit, W.; Piyakulpinyo, T.; Phonyiam, K.; Pongnumkul, S.; Chaovalit, P.; Theeramunkong, T. A survey on evaluation methods for chatbots. Proceedings of the 2019 7th International Conference on Information and Education Technology, 2019, pp. 111–119.
22. Goddard, M. The EU General Data Protection Regulation (GDPR): European regulation that has a global impact. *International Journal of Market Research* **2017**, *59*, 703–705.
23. Huckvale, K.; Prieto, J.T.; Tilney, M.; Benghozi, P.J.; Car, J. Unaddressed privacy risks in accredited health and wellness apps: a cross-sectional systematic assessment. *BMC medicine* **2015**.
24. Möller, S.; Raake, A. *Quality of experience: advanced concepts, applications and methods*; 2014.
25. Reiter, U.; Brunnström, K.; De Moor, K.; Larabi, M.C.; Pereira, M.; Pinheiro, A.; You, J.; Zgank, A. Factors influencing quality of experience. In *Quality of experience*; 2014; pp. 55–72.
26. Najjar, A.; Gravier, C.; Serpaggi, X.; Boissier, O. Modeling User Expectations & Satisfaction for SaaS Applications Using Multi-agent Negotiation. Web Intelligence (WI), 2016 IEEE/WIC/ACM International Conference on, 2016, pp. 399–406.
27. Najjar, A.; Serpaggi, X.; Gravier, C.; Boissier, O. Multi-agent systems for personalized qoe-management. Teletraffic Congress (ITC 28), 2016 28th International, 2016, Vol. 3, pp. 1–6.
28. Walgama, M.; Hettige, B. Chatbots: The next generation in computer interfacing–A Review **2017**.
29. Cahn, J. CHATBOT: Architecture, design, & development. *University of Pennsylvania School of Engineering and Applied Science Department of Computer and Information Science* **2017**.
30. Bentivoglio, C.; Bonura, D.; Cannella, V.; Carletti, S.; Pipitone, A.; Pirrone, R.; Rossi, P.; Russo, G. Intelligent Agents supporting user interactions within self regulated learning processes. *Journal of E-learning and Knowledge Society* **2010**, *6*, 27–36.
31. Haddadi, A. *Communication and cooperation in agent systems: a pragmatic theory*; Vol. 1056, 1996.
32. Chung, M.; Ko, E.; Joung, H.; Kim, S.J. Chatbot e-service and customer satisfaction regarding luxury brands. *Journal of Business Research* **2018**.
33. Calvaresi, D.; Calbimonte, A.I.J.P.; Schegg, R.; Fragniere, E.; Schumacher, M. The Evolution of Chatbots in Tourism: a Systematic Literature Review **2021**.
34. Żytniewski, M. Integration of knowledge management systems and business processes using multi-agent systems. *International Journal of Computational Intelligence Studies* **2016**.
35. Alencar, M.; Netto, J.M. Improving cooperation in Virtual Learning Environments using multi-agent systems and AIML. Frontiers in Education Conference (FIE), 2011, 2011.
36. Hettige, B.; Karunananda, A. Octopus: a multi agent chatbot **2015**.

37. Calvaresi, D.; Cesarini, D.; Sernani, P.; Marinoni, M.; Dragoni, A.F.; Sturm, A. Exploring the ambient assisted living domain: a systematic review. *Journal of Ambient Intelligence and Humanized Computing* **2016**, pp. 1–19.
38. Pereira, J.; Díaz, Ó. Using health chatbots for behavior change: a mapping study. *Journal of medical systems* **2019**, *43*, 135.
39. Brinkman, W.P. Virtual health agents for behavior change: Research perspectives and directions. Proceedings of the Workshop on Graphical and Robotic Embodied Agents for Therapeutic Systems, 2016.
40. Teixeira, A.R. Social Media and Chatbots use for chronic disease patients support: case study from an online community regarding therapeutic use of cannabis **2019**.
41. Lisetti, C.; Amini, R.; Yasavur, U. Now all together: overview of virtual health assistants emulating face-to-face health interview experience. *KI-Künstliche Intelligenz* **2015**, *29*, 161–172.
42. Schueller, S.M.; Tomasino, K.N.; Mohr, D.C. Integrating human support into behavioral intervention technologies: the efficiency model of support. *Clinical Psychology: Science and Practice* **2017**, *24*, 27–45.
43. Oh, K.J.; Lee, D.; Ko, B.; Choi, H.J. A chatbot for psychiatric counseling in mental healthcare service based on emotional dialogue analysis and sentence generation. 2017 18th IEEE International Conference on Mobile Data Management (MDM). IEEE, 2017, pp. 371–375.
44. van Heerden, A.; Ntinga, X.; Vilakazi, K. The potential of conversational agents to provide a rapid HIV counseling and testing services. 2017 international conference on the frontiers and advances in data science (FADS). IEEE, 2017, pp. 80–85.
45. Cheng, A.; Raghavaraju, V.; Kanugo, J.; Handrianto, Y.P.; Shang, Y. Development and evaluation of a healthy coping voice interface application using the Google home for elderly patients with type 2 diabetes. 2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC). IEEE, 2018, pp. 1–5.
46. Richards, D.; Caldwell, P. Improving health outcomes sooner rather than later via an interactive website and virtual specialist. *IEEE journal of biomedical and health informatics* **2017**, *22*, 1699–1706.
47. Richards, D.; Caldwell, P. Gamification to improve adherence to clinical treatment advice. *Health literacy: Breakthroughs in research and practice: Breakthroughs in research and practice* **2017**, *80*.
48. Bickmore, T.W.; Puskar, K.; Schlenk, E.A.; Pfeifer, L.M.; Sereika, S.M. Maintaining reality: Relational agents for antipsychotic medication adherence. *Interacting with Computers* **2010**, *22*, 276–288.
49. Tanner, A.E.; Mann, L.; Song, E.; Alonzo, J.; Schafer, K.; Arellano, E.; Garcia, J.M.; Rhodes, S.D. weCARE: A social media–based intervention designed to increase HIV care linkage, retention, and health outcomes for racially and ethnically diverse young MSM. *AIDS Education and Prevention* **2016**, *28*, 216–230.
50. Schmaltz, R.M.; Jansen, E.; Wenckowski, N. Redefining critical thinking: Teaching students to think like scientists. *Frontiers in Psychology* **2017**, *8*, 459.
51. Ramo, D.E.; Thrul, J.; Chavez, K.; Delucchi, K.L.; Prochaska, J.J. Feasibility and quit rates of the Tobacco Status Project: A Facebook smoking cessation intervention for young adults. *J of medical Internet research* **2015**, *17*.
52. Cole-Lewis, H.; Perotte, A.; Galica, K.; Dreyer, L.; Griffith, C.; Schwarz, M.; Yun, C.; Patrick, H.; Coa, K.; Augustson, E. Social network behavior and engagement within a smoking cessation Facebook page. *J of medical Internet research* **2016**, *18*.
53. Cheung, Y.T.D.; Chan, C.H.H.; Lai, C.K.J.; Chan, W.F.V.; Wang, M.P.; Li, H.C.W.; Chan, S.S.C.; Lam, T.H. Using WhatsApp and Facebook online social groups for smoking relapse prevention for recent quitters: A pilot pragmatic cluster randomized controlled trial. *J medical Internet research* **2015**, *17*.
54. Pechmann, C.; Pan, L.; Delucchi, K.; Lakon, C.M.; Prochaska, J.J. Development of a Twitter-based intervention for smoking cessation that encourages high-quality social media interactions via automessages. *J medical Internet research* **2015**, *17*.
55. Brixey, J.; Hoegen, R.; Lan, W.; Rusow, J.; Singla, K.; Yin, X.; Artstein, R.; Leuski, A. SHIHbot: A Facebook chatbot for sexual health information on HIV/AIDS. Proceedings of the 18th annual SIGdial meeting on discourse and dialogue, 2017, pp. 370–373.
56. Vita, S.; Marocco, R.; Pozzetto, I.; Morlino, G.; Vigilante, E.; Palmacci, V.; Fondaco, L.; Kertusha, B.; Renzelli, M.; Mercurio, V.; others. The'doctor apollo'chatbot: a digital health tool to improve engagement of people living with HIV. *J Int AIDS Soc* **2018**, *21*, e25187.
57. Angara, P.; Jiménez, M.; Agarwal, K.; Jain, H.; Jain, R.; Stege, U.; Ganti, S.; Müller, H.A.; Ng, J.W. Foodie fooderson a conversational agent for the smart kitchen. Proc. 27th Annual International Conference on Computer Science and Software Engineering, 2017, pp. 247–253.

58. Hsu, P.; Zhao, J.; Liao, K.; Liu, T.; Wang, C. AllergyBot: A Chatbot technology intervention for young adults with food allergies dining out. Proceedings of the 2017 CHI conference extended abstracts on human factors in computing systems, 2017, pp. 74–79.
59. Ghandeharioun, A.; McDuff, D.; Czerwinski, M.; Rowan, K. Towards understanding emotional intelligence for behavior change chatbots. 2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII). IEEE, 2019, pp. 8–14.
60. Ziebarth, S.; Kizina, A.; Hoppe, H.U.; Dini, L. A serious game for training patient-centered medical interviews. Advanced Learning Technologies (ICALT), 2014 IEEE 14th International Conference on, 2014, pp. 213–217.
61. Palanica, A.; Flaschner, P.; Thommandram, A.; Li, M.; Fossat, Y. Physicians' perceptions of chatbots in health care: cross-sectional web-based survey. *Journal of medical Internet research* **2019**, *21*, e12887.
62. Eysenbach, G.; Group, C.E.; others. CONSORT-EHEALTH: improving and standardizing evaluation reports of Web-based and mobile health interventions. *Journal of medical Internet research* **2011**, *13*, e126.
63. Kieffer, S.; Ko, A.; Ko, G.; Gruson, D. Six case-based recommendations for designing mobile health applications and chatbots. *Clinica Chimica Acta* **2019**, *493*, S29–S30.
64. Calvaresi, D.; Marinoni, M.; Dragoni, A.F.; Hilfiker, R.; Schumacher, M. Real-time multi-agent systems for telerehabilitation scenarios. *Artificial intelligence in medicine* **2019**, *96*, 217–231.
65. Palanca, J.; Terrasa, A.; Julian, V.; Carrascosa, C. SPADE 3: Supporting the New Generation of Multi-Agent Systems. *IEEE Access* **2020**, *8*, 182537–182549.
66. Füzéki, E.; Groneberg, D.A.; Banzer, W. Physical activity during COVID-19 induced lockdown: recommendations. *Journal of Occupational Medicine and Toxicology* **2020**, *15*, 1–5.