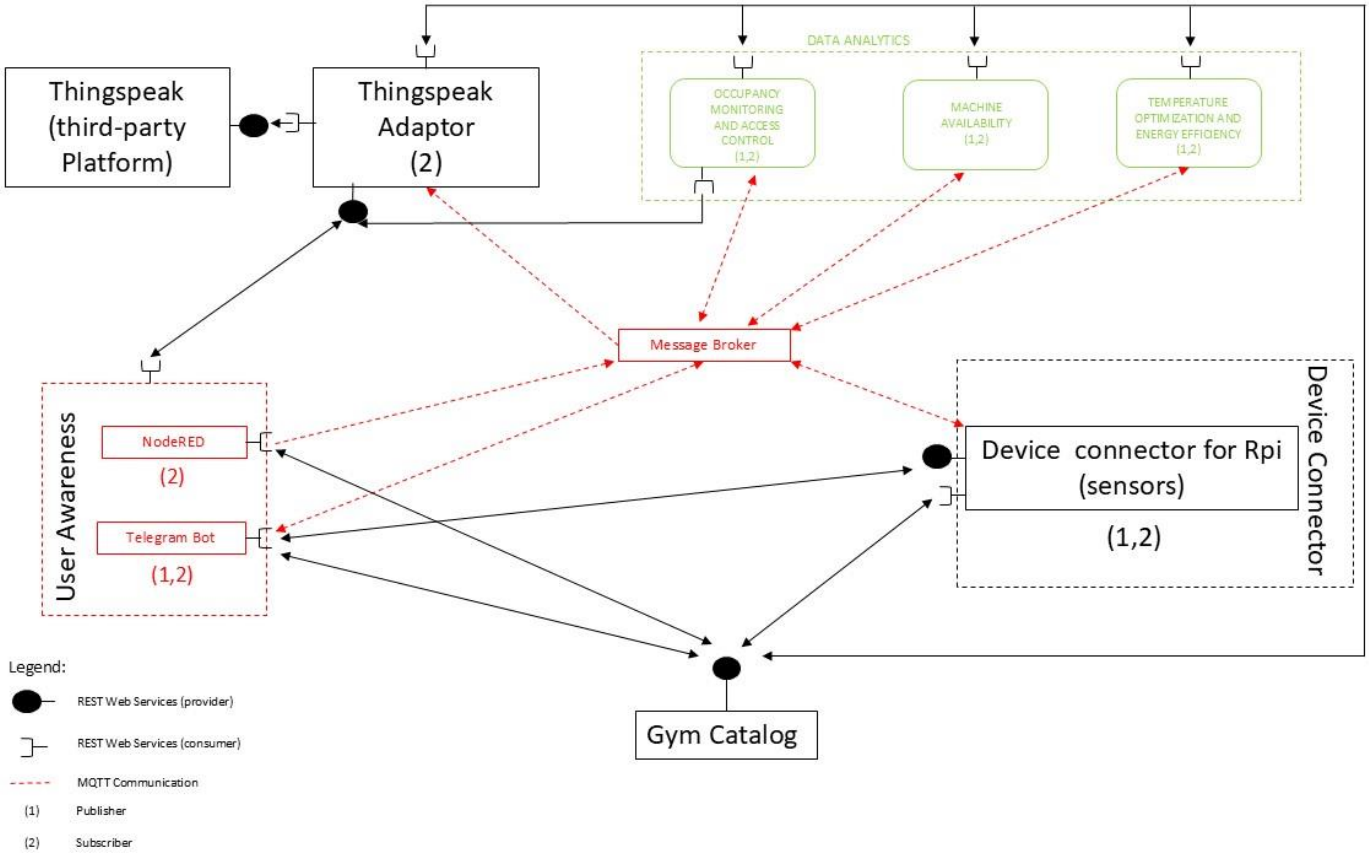
1. Name of Use Case

Name of the Use Case	**Gym Genius**
Version No.	v0.3
Submission Date	7/12/2023
Team Members (with student ids)	Yasmin Awed s320303, Augusta Castelli s320230, Shurui Chen s307957, Federico Villata s319922

2. Scope and Objectives of Function

Scope and Objectives of Use Case	
Scope	The purpose of the proposed IoT platform is to offer services for smart gym management.
Objective(s)	The IoT application Gym Genius is intended to optimize gym management, improve customer satisfaction, and guarantee optimal utilization of resources. It employs automated mechanisms, a variety of sensors, and a Telegram bot to facilitate user interaction.
Domain(s)	Smart Building, Smart Facilities
Stakeholder(s)	Gym owners/managers, gym staff and gym users/members
Short description	The goal of the presented IoT platform is to offer a smart solution to manage a gym environment and optimize customer experience. It incorporates a variety of features such as: • the possibility to check whether a particular type of machine is accessible via an infrared sensor for each machine; • the ability to track the gym's occupancy through a push button that opens the door for individual entry/exit; • the capacity to efficiently control, both remotely or in an automated way, the building's HVAC (Heating, Ventilation, and Air Conditioning), to minimize energy waste and encourage environmentally friendly behavior.

3. Diagram of Use Case



4. Complete description of the system

Gym Genius follows the microservices designing pattern and it makes use of two communication paradigms: request/response based on REST Web Services and publish/subscribe based on MQTT protocol. Ten actors have been identified in this context and presented in the following:

- The **Message Broker** offers a publish/subscribe method-based asynchronous communication exploiting the MQTT protocol.

- The **Gym Catalog** is composed of two parts: the **Service Catalog** and the **Resource Catalog**. The Service Catalog keeps track of all active services, while the Resource Catalog registers all devices within the system. Each service or device automatically registers itself with the appropriate catalog upon

initialization. The Gym Catalog provides details on all endpoints (e.g., REST Web Services and MQTT topics) associated with devices, resources, and services on the platform. Additionally, it offers configuration options for control strategies and applications. Services rely on the Service Catalog's REST Web Services to discover and communicate with other services by retrieving their endpoint information.

- The **Device Connector** integrates the sensors with the platform using Raspberry Pi boards. The Raspberry Pi is equipped with temperature and humidity sensors to provide environmental information about the building's condition, infrared sensors to detect the occupancy of specific machines, and push buttons to monitor the entrance and exits of the gym. It offers REST Web Services for obtaining all this information. Furthermore, the Raspberry Pi functions both as an MQTT publisher and subscriber. As a publisher, it transmits environmental data every five minutes, user presence information (when detected). As a subscriber, it receives actuation commands from other actors using the MQTT protocol (e.g., Temperature optimization and energy efficiency block) and on/off commands received from the Telegram bot to control connected appliances. To scale the platform, multiple Raspberry Pis can be deployed in different rooms (entrance, changing rooms, and activity rooms), each equipped with sensors tailored to their specific needs (temperature/humidity sensors, push buttons, infrared sensors, etc.).

- The **Occupancy monitoring and access control** serve as the backbone for monitoring customer presence within the gym, enabling real-time occupancy updates recovered from device connectors. Moreover, the microservice exploits algorithms to make predictions about the occupancy of the gym at different time intervals of the week to optimize gym operations and client experiences. Integration of MQTT and REST facilitates efficient data handling: it works as an MQTT subscriber to receive information on user’s accesses and exits, whereas it implements REST APIs to send and receive information and to/from the Gym Catalog and to receive historical data from the Thingspeak adaptor.

- The **Machine availability** block tracks the availability of specific gym machine types, showing which ones are being used or not. It works as an MQTT subscriber to receive information about the availability of each individual machine and it acts as an MQTT publisher to transmit Aggregated Machine Availability data, providing an overview of overall machine usage in the gym, which can be used by other services or platforms (eg. Telegram Bot). Additionally, it utilizes RESTful APIs to send and receive information to/from the Gym Catalog.

- The **Temperature optimization and energy efficiency** block gathers information from humidity and temperature sensors positioned all over the structure. To optimize HVAC (heating, ventilation, and air conditioning) usage and reduce energy waste during off-peak hours, it takes into consideration temperature variations concerning occupancy levels. In particular, it works both as an MQTT subscriber, to receive information on the gym’s environment and on the HVAC status, and as an MQTT publisher to send actuation commands to the Device Connector. Furthermore, it employs REST to send/receive data to/from the Gym Catalog.

- The **Thingspeak Adaptor** is an MQTT subscriber that receives measurements on environmental data and data analytics and uploads them to ThingSpeak through REST Web Services. Additionally, REST APIs are used to send and receive data to/from the Gym Catalog. Moreover REST APIs are also employed to send data to obtain historical data from ThingSpeak, and can be retrieved by other services or platforms (eg. NodeRed).

- **Thingspeak** is a third-party software (https://thingspeak.com/) that provides REST Web Services to allow the Thingspeak Adaptor to retrive historical data. It is an open-data platform for the Internet of Things to store, post-process, and visualize data (through plots).

- **NodeRED** is a dashboard that employs Raspberry Pi's REST Web Services to retrieve data from the Device Connector, to imports historical data from the Thingspeak Adaptor and to send/receive data to/from the Gym Catalog. Furthermore, it works as an MQTT subscriber to retrive even more data. It is used to plot and visualize.

- **Telegram Bot** is a service that connects the proposed infrastructure to the cloud-based instant messaging platform Telegram. It makes use of the REST Web Services offered by the Device Connector to retrieve measurements from the devices of the platform and to send/receive data to/from the Gym Catalog.

Additionally, it functions both as an MQTT publisher and an MQTT subscriber, allowing users to send actuation commands (e.g., temperature ranges) and retrieve alerts in real time.

5. Hardware components (only among those we can provide)

No additional devices are needed.