iMuseum Architectural Evaluation

Introduction and System Description

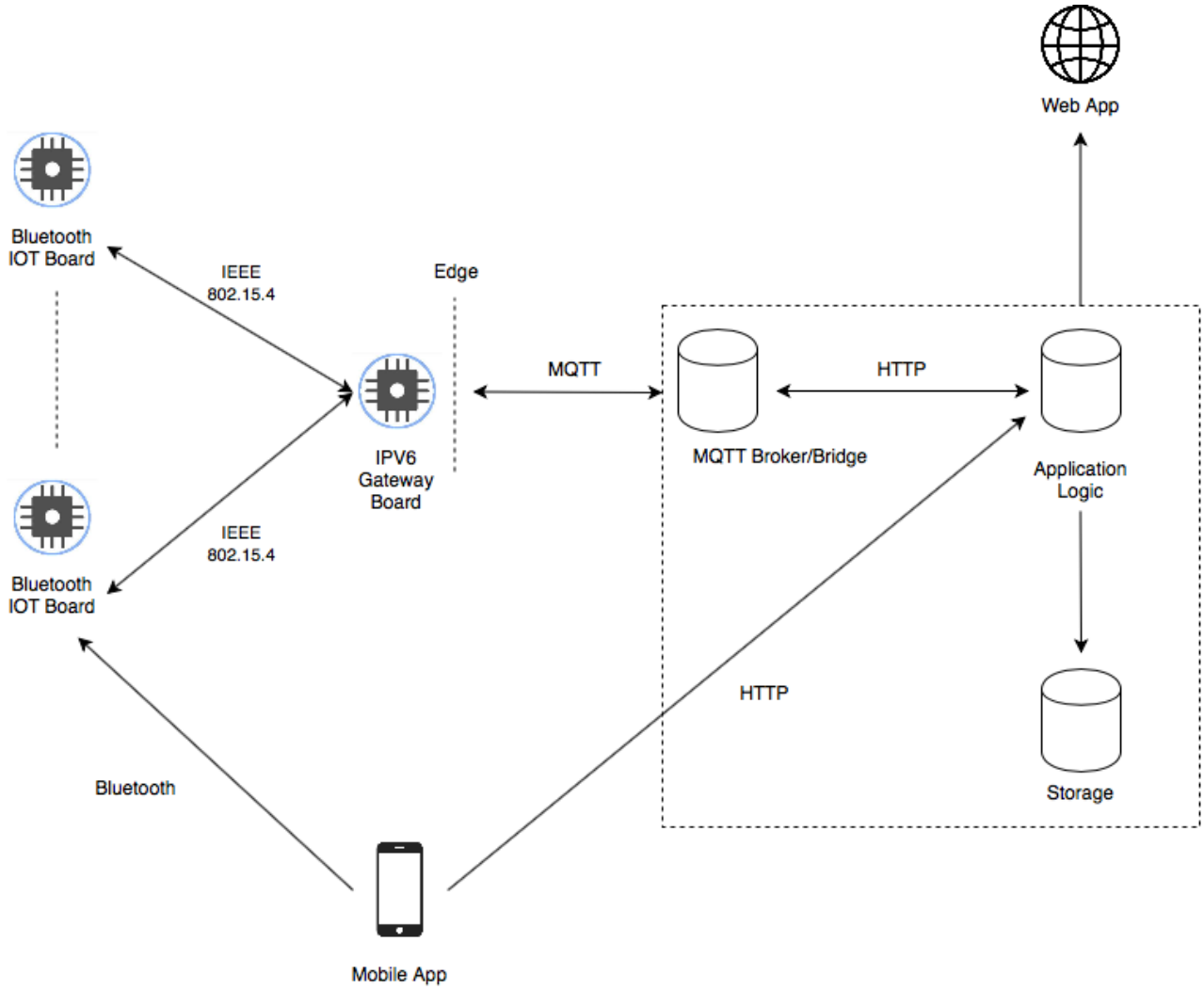
iMuseum is a distributed system to enhance the experience for a visitor in a museum: it uses IOT devices placed on each art piece to provide informations to the user through a mobile app using the bluetooth beacon mode.

Once the visit is over, the user will get a report of all the pieces seen in the visit.

The system also provides to the curator of the museum/exhibition all the informations about the devices status, a record of all the visitors along with the path and pieces seen inside the museum and all the informations about the sensors status to help notify the maintenance about possible malfunctions.

iMuseum Architecture

Here is shown a visual representation of the system's architecture:



starting from the right we have a variable number of IOT boards (one on each piece) that communicates to a mobile app using the bluetooth beacon mode: once a visitor comes near a piece, the mobile application will ask to the user if he/she want to know more informations about it, if it's the case, the system will automatically record the preference and provide a comprehensive list of informations about the piece.

All the informations about the user and the visits can be easily seen by the curator of the museum using a web app that also provides informations about the status of the sensors, notifying the administrator about possible failures of one or more of the sensors.
The data comes periodically from the beacons to a gateway board with COAP support (one for every museum) and then sent back to a COAP bridge to finally comes over HTTP to the Application Logic tier trough REST endpoints, to be stored inside the Storage.

Hardware Components

Bluetooth IOT Board

Category: IOT

The Bluetooth IOT board is in charge of beaming data using the bluetooth beacon mode (https://en.wikipedia.org/wiki/Bluetooth_low_energy_beacon), providing the informations of the piece where it's placed, it also communicate to a COAP gateway, data about it's status through the IEEE_802.15.4 protocol (https://en.wikipedia.org/wiki/IEEE_802.15.4).

The board that can best handle both the data transmission is the nRF52840DK (https://www.nordicsemi.com/Software-and-Tools/Development-Kits/nRF52840-DK)
that can handle both bluetooth and low-data wireless transmission.

IPV6 Gateway Board

Category: Edge Component

The IPV6 gateway board (https://www.st.com/en/evaluation-tools/stm32-nucleo-boards.html/) receives all the informations of local network of the museum and forward them to the internet. It also run a RIOT application to detect incoming data from the sensors status, scattered in the exhibition.

The real-world device deployed in the net uses a STM nucleo board(https://www.st.com/en/evaluation-tools/p-nucleo-wb55.html) pared with a ethernet module (https://www.ebay.it/itm/Arduino-Compatibile-Modulo-scheda-ethernet-rete-ENC28J60-LAN-Atmel-STM-ARM-/222596170068).

Smartphone (Mobile App)

Category: End-user component

The visitor will use his/her own smartphone (it can either run iOS or Android) obtain the identifier of a piece, using bluetooth, and uses it to obtain all the related informations from the CLOUD structure.

MOQTT Broker/Bridge

Category: Cloud Component

The MQTT Bridge is in charge of receiving data from the both from gateway and the application logic and forward them to the right subscribers.

Application Logic

Category: Cloud Component

The Application Logic tier is in charge to handle all the request coming via REST protocol, mainly to all the resources/data inside the storage, it's deployed inside the firebase cloud infrastructure (https://firebase.google.com).

Storage

Category: Cloud Component

Is in charge of the persistence of the data about the sensors status, visitors identities, pieces images, pieces documents etc.. it also deployed inside the FCI (https://firebase.google.com)

Web App

Category: End-user component

It displays all the informations to a curator of a museum/exhibition about all the visits, (in particular the preferred routes in each visit), the status of all the IOT sensors and finally a curator can create/ delete sensors inside the museum.

Software Components

RIOT OS

RiotOS (https://www.riot-os.org) is operative system used as base to write firmwares for a variety of embedded boards in C language, it's used to send data from the bluetooth nodes and from the COAP gateway to the COAP bridge.

MQTT

It's a lightweight publish-subscriber protocol(http://mqtt.org). It's used both to forward the data coming from the boards, to the Application logic and the other way around.

Ponte

Eclipse framework(https://www.eclipse.org/ponte/) that receives MQTT data and forward them to the cloud using REST, used both to bridge the gap between the edge of the IOT system and CLOUD infrastructure and ensure a two way communication channel between the Application logic and the boards.

REST

It's an architecture that let web service communicate, it's the the base form of communication(over the HTTP protocol) between all the elements inside the cloud and the end-user presentation elements (https://en.wikipedia.org/wiki/Representational_state_transfer).

Xamarin Forms

It a framework that translate C# (OOP Language) at runtime to a native mobile device (can either be iOS or Android), used as a base to build the presentation to the user of the mobile app (https://docs.microsoft.com/it-it/xamarin/xamarin-forms/).

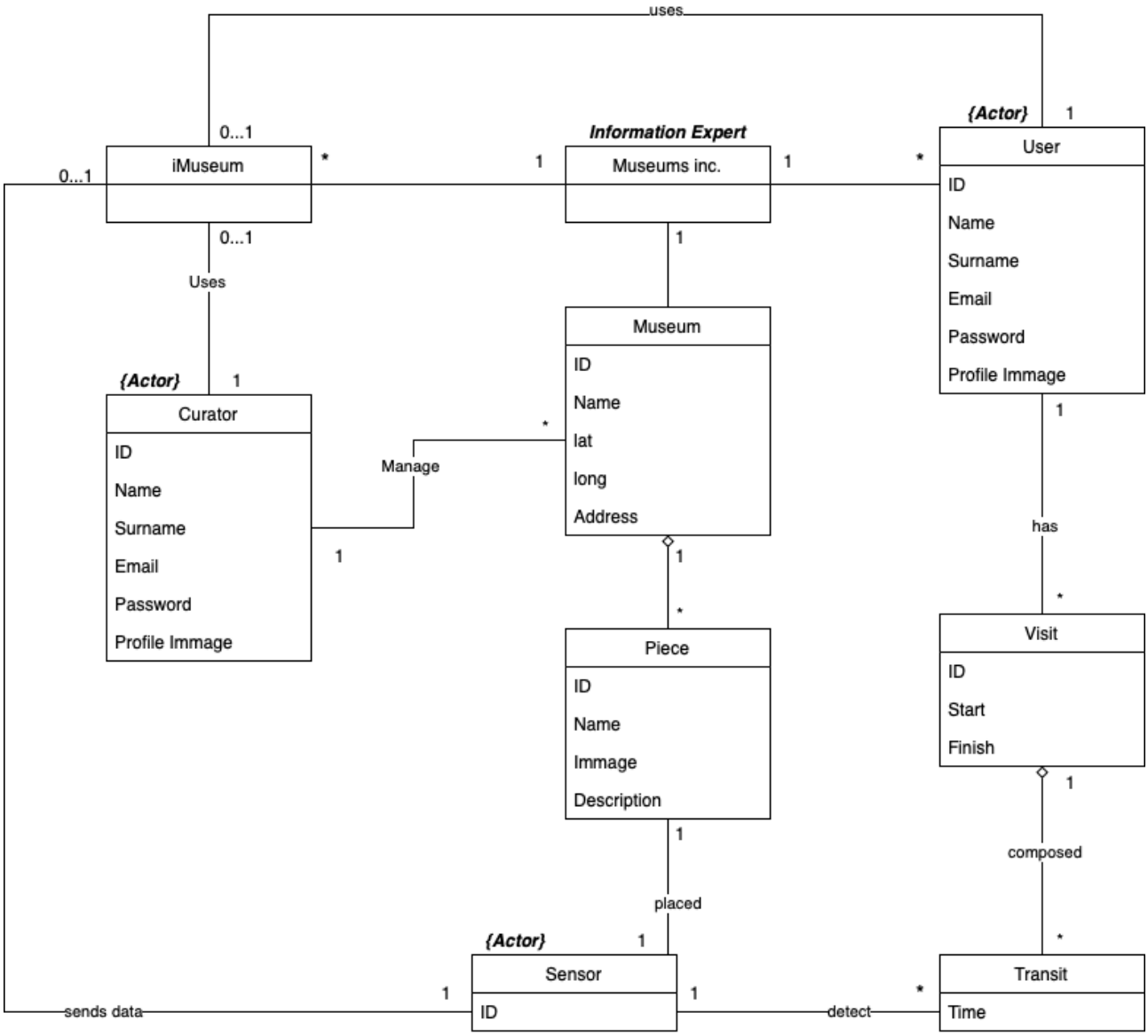
Javascript

High-level programming language used both in the application logic and the presentation to user in the form of the web app(https://en.wikipedia.org/wiki/JavaScript).

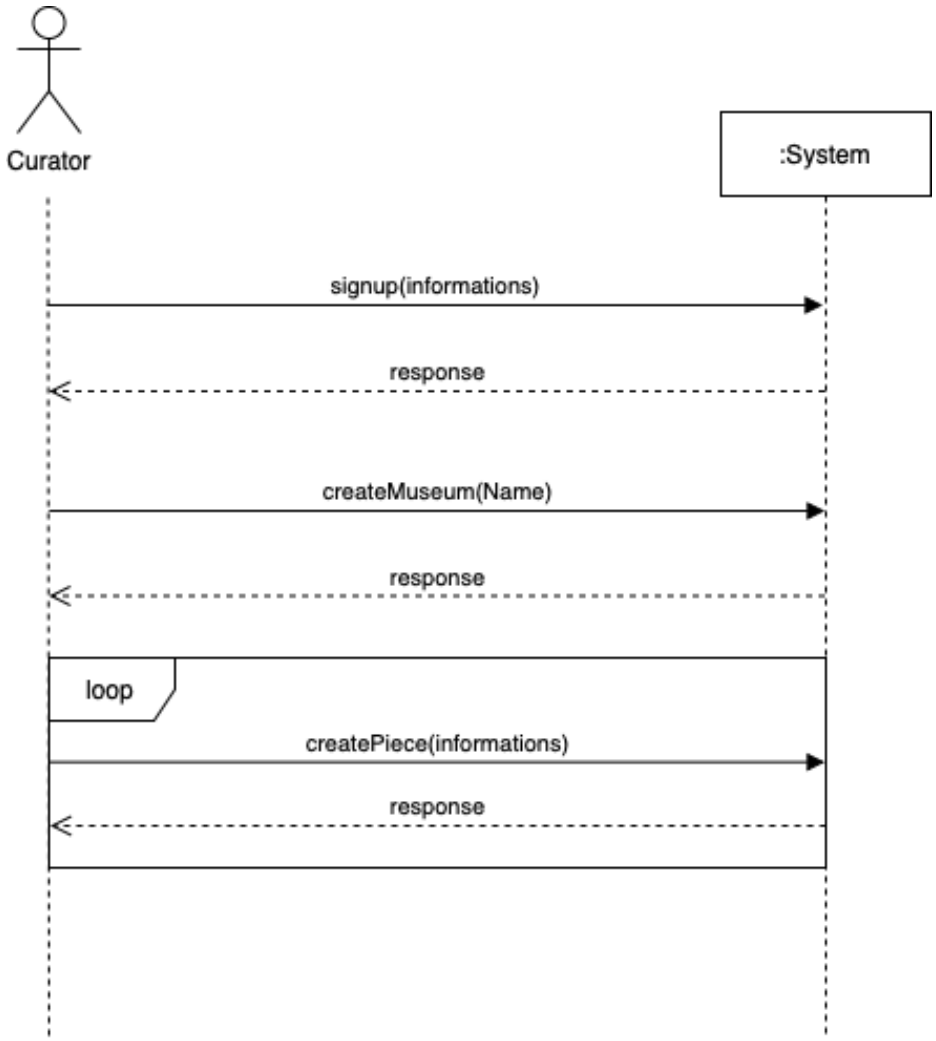
React Web

Framework to build interfaces combining HTML for the GUI and javascript for the scripting logic, used to build the web application to consult informations and data for the web app (https://en.wikipedia.org/wiki/JavaScript).

Analysis


uses
0...1
iMuseum
*
1
Information Expert
Museums inc.
1
*
{Actor}
1
User
ID
Name
Surname
Email
Password
Profile Immage
0...1
0...1
Uses
1
Museum
ID
Name
lat
long
Address
{Actor}
1
Curator
ID
Name
Surname
Email
Password
Profile Immage
*
Manage
1
1
1
has
*
Visit
ID
Start
Finish
*
1
Piece
ID
Name
Immage
Description
1
composed
placed
{Actor}
1
*
1
Sensor
ID
1
Transit
Time
sends data
detect

UC1: Signup of a Curator



CO1: singup

Operation: signup(informations)

Use case: UC1

Pre-Conditions: none

Post-Conditions:

- It has been created a link between the curator and the Museum
- It has been created a link between the current curator and the System.

CO2: createMuseum

Operation: createMuseum(informations)

Use case: UC1

Pre-Conditions:

- The curator has logged in.

Post-Conditions:

- It has been created a link between the Museum and the curator.

CO3: createPiece

Operation: createPiece(informations)

Use case: UC1

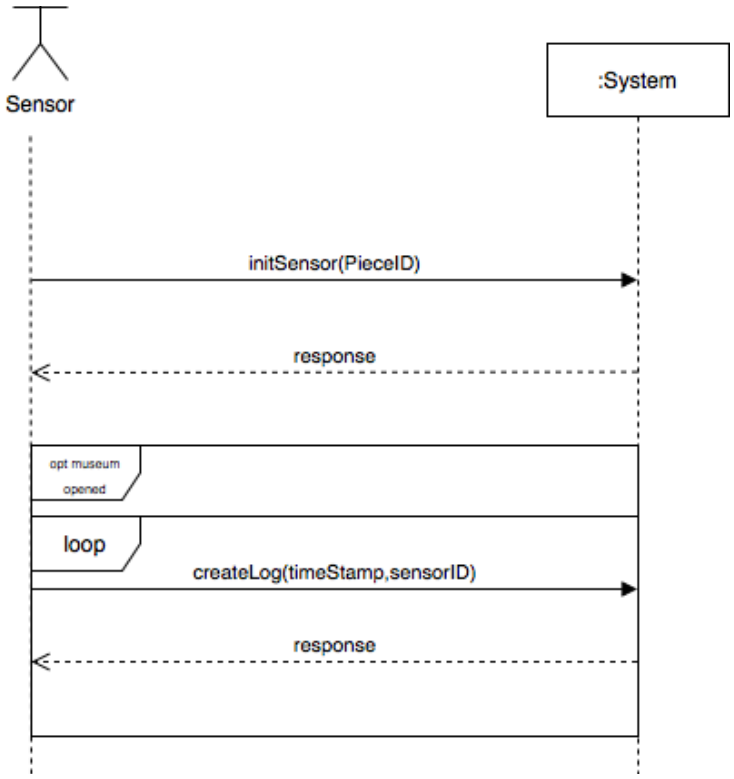
Pre-Conditions:

The curator has logged in.

Post-Conditions:

- It has been created a link between the piece and the Museum.

UC2: Initialization of a sensor



CO1: initSensor

Operation: initSensor(PieceID)

Use case: UC2

Pre-Conditions: none

Post-Conditions:

- It has been created a link between the sensor and the piece associate to the ID.

CO2: createLog

Operation: createLog(timestamp, sensorID)

Use case: UC2

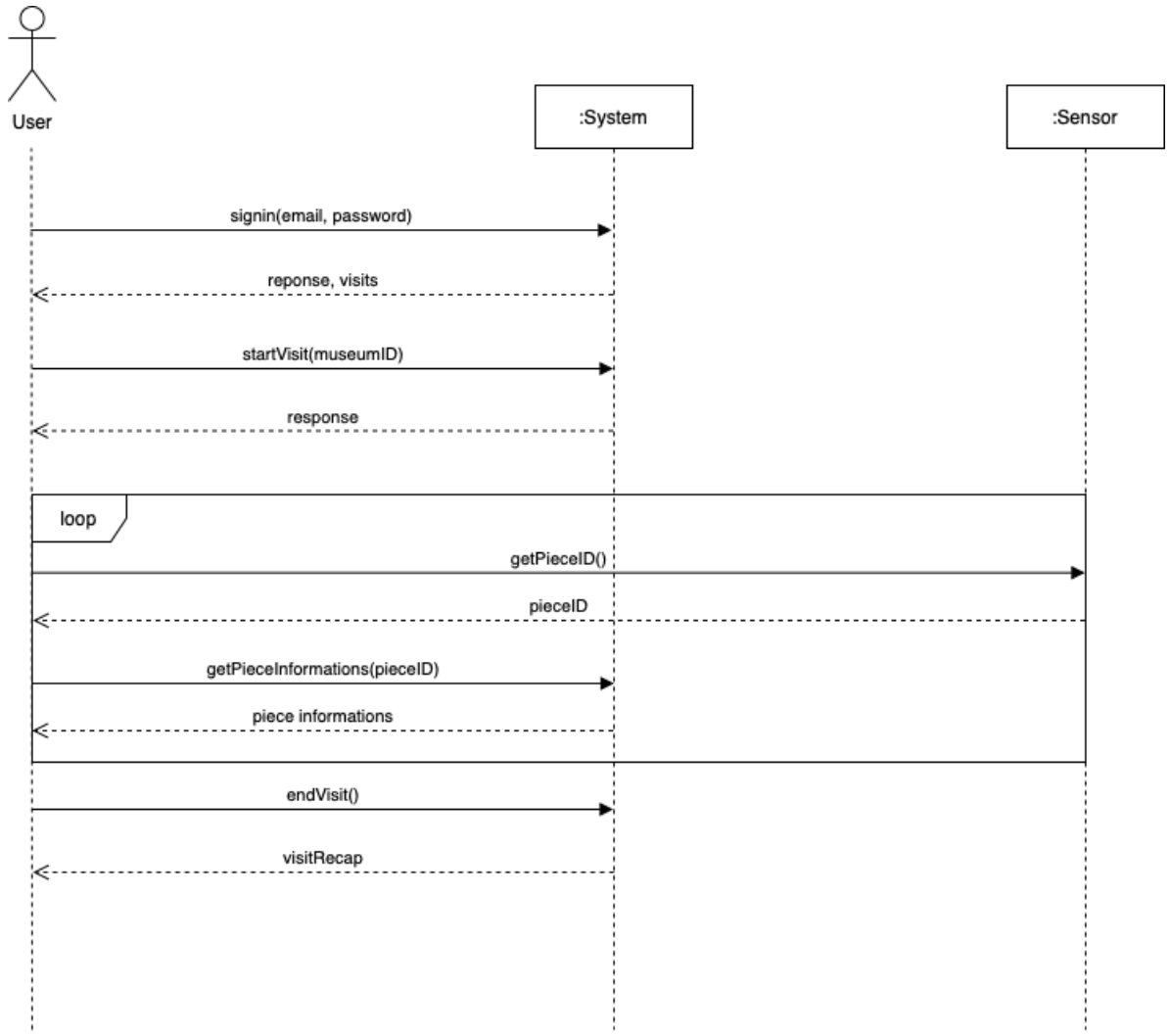
Pre-Conditions:

- The sensor has been initialized.

Post-Conditions:

- It has been created a link between the log and the sensor where the log was generated from.

UC3: Start a visit



CO:1 signin

Operation: signin(email,password)

Use case: UC3

Pre-Conditions: none

Post-Conditions:

- It has been created a link between the System and the user associate to the ID.

CO2: startVisit

Operation: startVisit(museumID)

Use case: UC3

Pre-Conditions: none

Post-Conditions:

- It has been created a new instance of visit.

CO3: getPieceID

Operation: getPieceID()

Use case: UC3

Pre-Conditions: none

Post-Conditions: none

CO4: getPieceInformations

Operation: getPieceInformations(pieceID)

Use case: UC3

Pre-Conditions: none

Post-Conditions:

- It has been created a new instance of transit.
- It has been created a link between transit and piece.
- It has been create a link between transit and visit.

CO5: endVisit

Operation: endVisit()

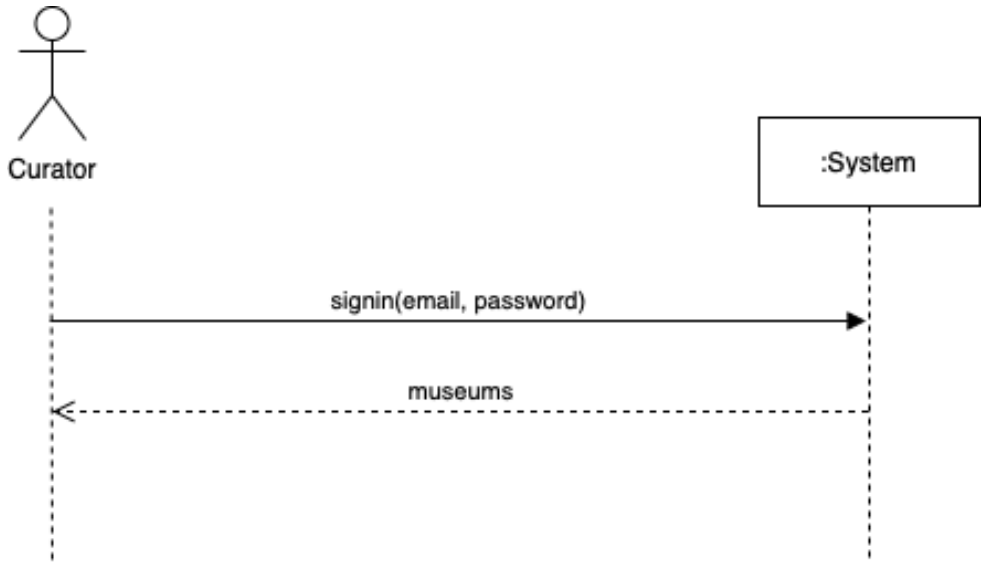
Use case: UC3

Pre-Conditions: none

Post-Conditions:

- It has been created a link the visit and the user.

UC4: Get Museums Statistics



CO1: signin

Operation: signin(email,password)

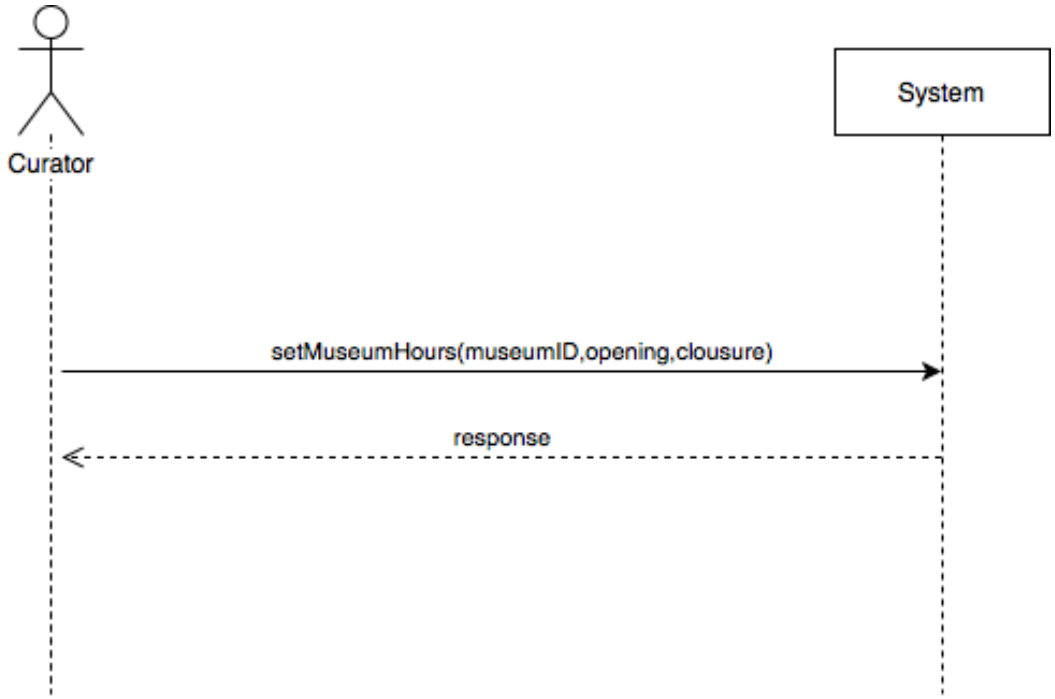
Use case: UC4

Pre-Conditions: none

Post-Conditions:

- It has been created a link between the System and the museum managed by the curator.

UC5: Setting Museum's opening times



CO1: setMuseumHours

Operation: setMuseumHours(museumID,opening,closure)

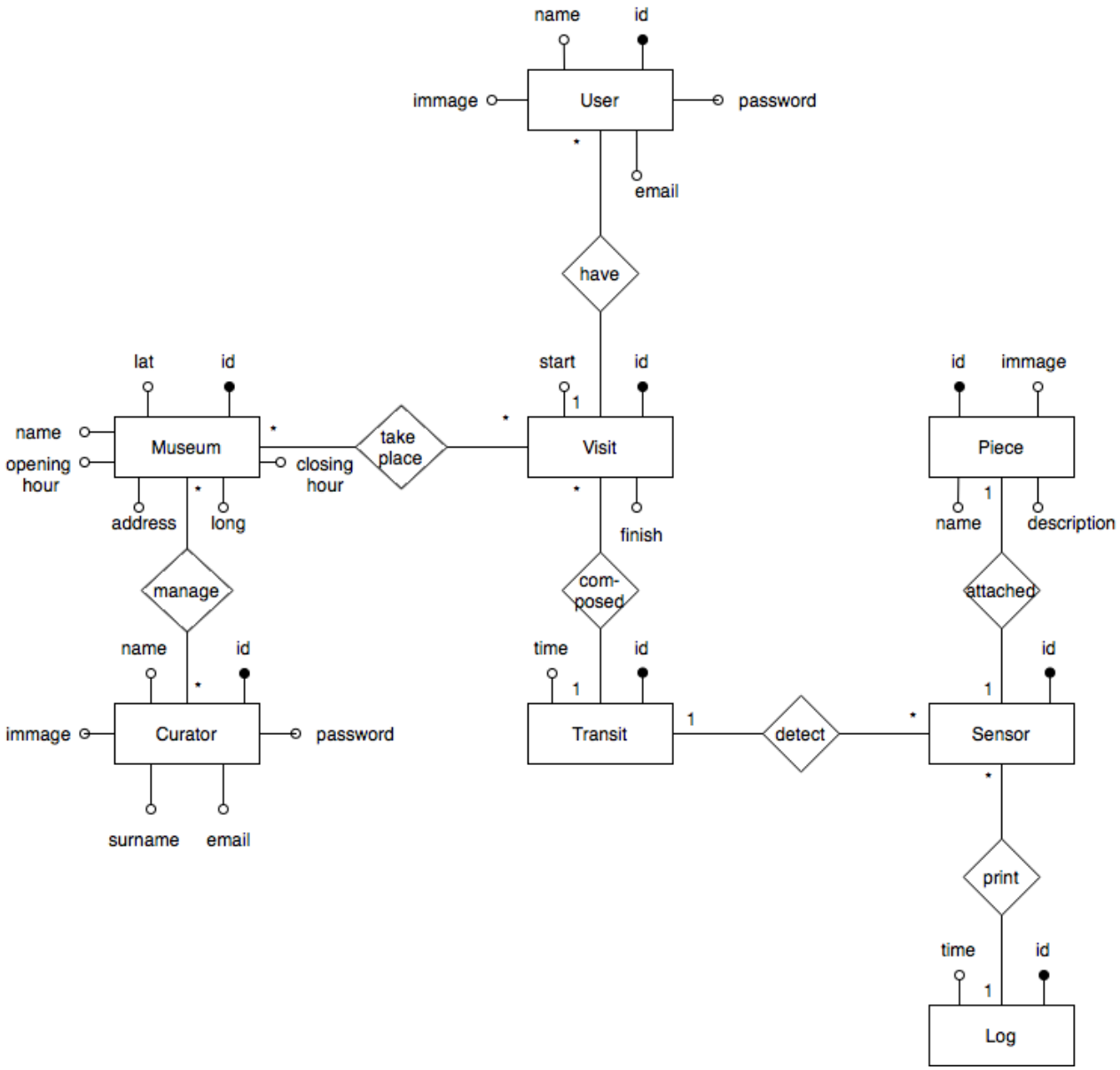
Use case: UC5

Pre-Conditions: none

Post-Conditions:

- It has been created a link between the museum based on the ID provided and the opening and closure hours

ER Model


name
id
immage
User
password
email
*
have
lat
id
start
id
id
immage
name
Museum
opening hour
closing hour
take place
Visit
Piece
address
long
finish
name
description
manage
com-posed
attached
name
id
time
id
id
immage
Curator
password
Transit
detect
Sensor
surname
email
print
time
id
Log