INFO GENERALI

Per facilitare, possiamo assumere un solo paziente per stanza e che id paziente = id stanza in cui si trova

Data format

Per saturazione e battito cardiaco paziente

{

“bn” : id stanza  
“bt” : timestamp  
“e” : [{ “n”:”perfusion index”, “u” : “”, “v” : },  
{ “n”:”saturation” , “u” : “” , “v” : },  
{ “n”:”pulse rate” , “u” : “” , “v” : }]}

Per temperatura paziente :

{

“bn” : id stanza

“bt” : timestamp

“e” : [{ “n”:”temperature”, “u” : “C”, “v” : 22.5 }

]}

Per stanza : (decidere se separarli o tenerli unici, in base al fatto se le operazioni rispettive sono svolte con la stessa frequenza ){

“bn” : id stanza

“bt” : timestamp

“e” : [{ “n”:”temperature”, “u” : “C”, “v” : 22.5 }

{ “n”:”light” , “u” : “integer” , “v” : 0 }]}

Contenuto file config.json

Città dell’ospedale  
Message broker e porta  
Base topic per paziente,stanza paziente e stanza comunre  
lista di chat id telegram, a cui mandare info  
lista di device attivi  
lista di servizi attivi  
lista dei pazienti  
lista delle stanze con pazienti dentro, e le rispettive temperature desiderate  
lista delle stanze comuni, e le loro rispettive temperature desiderate  
scheduling orario stanze comuni  
scheduling orario stanze paziente

Strutture base topic

Common Room : group01/IoTProject/CommonRoom/<id>  
Patient temperature : group01/IoTProject/Patient/Temperature/<id>  
Patient saturation and heartbeat : group01/IoTProject/Patient/pulsation/<id>  
Patient Room : group01/IoTProject/PatientRoom/<id>  
Patient Alarm : group01/IoTproject/PatientAlarm/<id>  
Common Room Command : group01/IoTproject/CommonRoomCommand/<id>  
Patient Room Command : group01/IoTproject/PatientRoomCommand/<id>

CLASSI

Patient Monitor

Una istanza per ogni paziente aggiunto.

ATTRIBUTI

(aggiungere attributi base)

METODI

Init : costruttore, inizializzare attributi.  
Start () : registra servizio, si memorizza base topic di input ed output, prende tutte le info necessarie dal catalog.  
setPatientId() : chiede in input l’id del paziente, fa la subscribe al topic.  
notify(arg1,arg2,msg ) : riceve messaggio con i dati del paziente, effettua i controlli. In caso di dati errati chiama la funzione ‘sendAlarm’ che invia l’allarme tramite una publish, ricevendo in input il messaggio.  
sendAlarm(msg) : fa la publish del messaggio ricevuto.  
OSS Nel main far partire anche il thread ( dopo il metodo start) che fa la PUT, tenendo aggiornato il servizio nel catalog

Light Patient Room Monitor

Una istanza per ogni stanza in cui c’è paziente.

ATTRIBUTI

(aggiungere attributi base)

METODI

Init : costruttore, inizializzare attributi.  
Start () : registra servizio (con una richiesta POST), si memorizza base topic di input ed output, prende tutte le info necessarie dal catalog.  
setPatientRoomId() : chiede in input (riga di comand) l’id della stanza, fa la subscribe al topic.  
notify(arg1,arg2,msg ) : riceve messaggio con i dati della stanza, effettua i controlli. Invoca ‘sendAlarm’ in caso ci sia bisogno di inviare un comando  
sendAlarm(msg) : invia il comando di attuazione.  
OSS Nel main far partire anche il thread ( dopo il metodo start) che fa la PUT, tenendo aggiornato il servizio nel catalog

Telegram Bot

Una singola istanza.

ATTRIBUTI

.lista dei chat id a cui mandare notifiche

METODI

Init( )   
Start( ) , registarsi al catalog, prendere topic e fare subscribe  
On\_chat\_messagge ( ) : in base al messaggio ricevuto, vede se infermiere vuole registrare nuovo paziente o se vuole settare  
Notify( ) : riceve l’alert e lo invia in chat a tutti i chat ID (memorizzati in una lista) che ha come attributo  
OSS Nel main far partire anche il thread ( dopo il metodo start) che fa la PUT, tenendo aggiornato il servizio nel catalog

Temperature common room monitor

Una istanza per ogni stanza comune, identificata da un id.

ATTRIBUTI  
scheduling orario

(aggiungere attributi base)

METODI

Init : costruttore, inizializzare attributi.  
Start () : registra servizio (con una richiesta POST), si memorizza base topic di input ed output, prende tutte le info necessarie dal catalog.  
setCommonRoomId() : chiede in input (riga di comando) l’id della stanza, fa la subscribe al topic.  
notify(arg1,arg2,msg ) : riceve messaggio con i dati della stanza, effettua i controlli. Invoca ‘sendAlarm’ in caso ci sia bisogno di inviare un comando  
sendCommand(msg) : invia il comando di attuazione.  
OSS Nel main far partire anche il thread ( dopo il metodo start) che fa la PUT, tenendo aggiornato il servizio nel catalog

Temperature patient room monitor

Una istanza per ogni stanza del paziente, identificata da un id.

ATTRIBUTI

Scheduling orario

(aggiungere attributi base)

METODI

Init : costruttore, inizializzare attributi.  
Start () : registra servizio (con una richiesta POST), si memorizza base topic di input ed output, prende tutte le info necessarie dal catalog.  
setCommonRoomId() : chiede in input (riga di comando) l’id della stanza, fa la subscribe al topic.  
notify(arg1,arg2,msg ) : riceve messaggio con i dati della stanza, effettua i controlli. Invoca ‘sendAlarm’ in caso ci sia bisogno di inviare un comando  
sendCommand(msg) : invia il comando di attuazione.  
OSS Nel main far partire anche il thread ( dopo il metodo start) che fa la PUT, tenendo aggiornato il servizio nel catalog

Fibrillation

Una istanza per ogni paziente, identificato da un id.

ATTRIBUTI

(aggiungere attributi base)

METODI

Init : costruttore, inizializzare attributi.  
Start () : registra servizio (con una richiesta POST), si memorizza base topic di input ed output, prende tutte le info necessarie dal catalog.  
setPatientId() : chiede in input (riga di comando) l’id del paziente, fa la subscribe al topic.  
notify(arg1,arg2,msg ) : riceve messaggio con i dati del paziente, effettua i controlli. Invoca ‘sendAlarm’ in caso ci sia bisogno di inviare un allarme di fibrillazione  
sendAlarm(msg) : invia il comando di attuazione.  
OSS Nel main far partire anche il thread ( dopo il metodo start) che fa la PUT, tenendo aggiornato il servizio nel catalog

RaspBerry emulator

Una istanza per simulare i vari raspberry.

ATTRIBUTI

Lista di pazienti per cui simulare dati  
Lista di stanze in cui ci sono pazienti  
Lista di stanze comuni

METODI

emulateCommonRoom( ) : ogni n secondi simula un dato per le stanze comuni, che sono lette dal catalog  
emulatePatientData( ) : ogni m secondi simula un dato per i pazienti, che sono lette dal catalog  
emulatePatientRoom( ) : ogni k secondi simula dati per le stanze dei pazienti, che sono lette dal catalog  
start () : si memorizza base topic di input ed output, prende tutte le info necessarie dal catalog ( come ad esempio lista iniziale di pazienti e stanze), fa subscribe gerarchica (col +)  
notify( ) : gestice il messaggio http ricevuto, che rappresenta un comando di attuazione  
OSS Nel main far partire anche il thread ( dopo il metodo start) che fa la PUT, tenendo aggiornato servizi e devices, magari riempendo anche le liste di pazienti e stanze per cui devo simulare i dati

Device and registry system

Una singola istanza, che gira sul server locale.  
  
ATTRIBUTI

File catalog, dizionario con tutte le info definite prima

METODI

Mette a disposizione i metodi GET, PUT, UPDATE e DELETE per svolgere le operazioni. Va stabilito come deve essere l’ url di ciascuna richiesta. Inoltre deve, ogni tot minuti, eliminare dalla lista di servizi e device quelli che non hanno fatto l’update

-Get  
città,pazienti,info su mqtt, servizi,dispositivi,topics,stanze,scheduling orari,telegram chat id

-Update  
device o servizi, aggiornando timestamp  
-Delete  
eliminare paziente   
-Post  
aggiungere paziente,servizi,devices.