*HOW TO MAKE ALL SCALABLE?*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PROPOSTE: gestione della registrazione/modifica/eliminazione di utenti-devices-stanze tramite la creazione di clients che fanno delle richieste al Catalog

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Creare un client ***Administrator.py*** che permette di registrare e aggiornare gli users/devices/rooms

questo client richiama tre diverse funzioni:

1. ***UserRegister.py***

che permette di registrare gli utenti:

* *POST catalog/ID\_utente/new*

modificare gli utenti esistenti:

* *PUT catalog/ID\_utente/change\_role catalog/ID\_utente/change\_assigned\_rooms*

eliminare:

* *DELETE catalog/ID\_utente/delete*

"userList": [

{"userID": "U\_001", "role": "worker", "roomIDs": ["R\_001", "R\_003"], **“chatID”: ???**},

{"userID": "U\_002", "role": "owner", "roomIDs": ["R\_002"]}

]

1. ***DeviceRegister.py***

che permette di registrare nuovi dispositivi:

* + POST catalog/ID\_stanza/ID\_device/new

aggiornare gli esistenti:

* + PUT catalog/ID\_stanza/ID\_device/update\_name
  + catalog/ID\_stanza/ID\_device/change\_meas\_type
  + catalog/ID\_stanza/ID\_device/change\_service\_details
  + catalog/ID\_stanza/ID\_device/change\_topic
  + catalog/ID\_utente/change\_role
  + catalog/ID\_utente/change\_assigned\_rooms
  + catalog/ID\_stanza/change\_product\_type

eliminarli:

* + DELETE catalog/ID\_stanza/ID\_device/delete

"roomList": [

{"roomID": "R\_001",

"devicesList": [

{"deviceID": "D\_001", "deviceName": "R1P1", "measureType": "AAA", **“field”: “field1”**,

"availableServices": ["MQTT", "REST"],"servicesDetails": [

{"serviceType": "MQTT", "serviceIP": "ciao11", "topic": ["mio/prova1","mio/prova11"]},

{"serviceType": "REST", "serviceIP": "ciao12", "topic": []}], **“channelID”: 134252, “api\_key\_read”: ZVBAO2QDON8B19X0, “api\_key\_write”: S6ULMDXZPCVFBR0H}, …]**

DUE OPZIONI PER **THINGSPEAK**: (IN OGNI CASO IL CATALOG ANDREBBE MODIFICATO COME IN ROSSO SEGNATO)

1. (NON MI PIACE E NON CREDO SIA FATTIBILE PERCHE’ ILARIA MI HA DATO LA NOTIZIA DI AVERE SOLO 4 CANALI A DISPOSIZIONE) al momento della registrazione di un device bisogna assegnare un canale thingspeak quindi bisogna crearlo e la risposta alla creazione (che si fa con un SEMPLICE POST usando l’api\_key del profilo e il nome “new channel”) implica il salvataggio delle api\_key quindi bisogna aggiungere due campi nel Catalog per ogni device che riguardano due URL per fare il WRITE e il READ del canale e sarebbero due URL tipo: <https://api.thingspeak.com/update?api_key=ZVBAO2QDON8B19X0> (WRITE) <https://api.thingspeak.com/channels/1321136/feeds.json?api_key=S6ULMDXZPCVFBR0H> (READ)

1. PREFERISCO -> Forse conviene assegnare un canale Thingspeak ad ogni stanza (lo si crea al momento di registrazione di una nuova stanza automaticamente) perchè c'è un limite di canali e poi ogni "fieldx" usarlo per ogni sensore della stanza. In corrispondenza di ogni device si dà un campo (field1, field2, field3...) -> va inserito nel Catalog e ogni stanza ha il suo channelID, e le 2 apy\_key per read e write: (modifiche in rosso sul Catalog per implementare ciò) mentre ogni device ha il suo “field”

<https://api.thingspeak.com/update?api_key=ZVBAO2QDON8B19X0&field1=0> (WRITE A CHANNEL FIELD) <https://api.thingspeak.com/channels/1321136/fields/1.json?api_key=S6ULMDXZPCVFBR0H> (READ A CHANNEL FIELD)

N.B: Ogni volta che si registra un nuovo device bisogna assegnargli un field nel canale assegnato alla stanza per salvarne i dati e ogni volta che si rimuove bisogna ripulire il field su Thingspeak. Ogni volta che si registra una stanza bisogna creare in automatico un channel su Thingspeak

1. ***RoomRegister.py***

che permette di registrare nuove stanze:

* POST catalog/ID\_stanza/new

di eliminarle:

* DELETE catalog/ID\_stanza/ID\_device/delete

richiama anche il DeviceRegister.py per l'inserimento dei devices

N.B: tutti questi Registers devono anche consentire la funzione di “updating” nel senso di aggiornarsi continuamente nel Catalog

N.B: Gestione degli inserimenti dei nuovi dispositivi/utenti/stanze. Potrebbero anche essere convertiti in un client tramite Telegram come voleva fare Riccardo, ma qui propongo una versione base

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PROPOSTE SUL TELEGRAM BOT E SERVICES

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

L’amministratore crea due Telegram Bot separati.

1. BOT PER GLI OWNERS
2. BOT PER I WORKERS

Cosa possono fare gli Owners?

* Vedere la lista delle stanze attive (sarebbe carino riuscire a selezionare da Telegram la stanza, ma non so come gestire il fatto che deve prendere il numero delle stanze dal Catalog e fare tanti ‘bottoncini’ quante sono le stanze)
* Vedere tutti i dispositivi in funzionamento
* Vedere tutti gli utenti e le assegnazioni alle varie stanze
* Vedere le statistiche complessive di tutti i sensori di temperatura e umidità e di tutte le stanze e i grafici su Thingspeak e gli ultimi aggiornamenti dei devices
* Vedere le entrate e le uscite complessive dei prodotti durante il giorno
* Vedere le quantità di prodotti nel magazzino e la tipologia

Cosa possono fare i workers?

* Ricevere le notifiche in caso di guasti o anomalie nelle proprie stanze (valutare cosa si riesce a fare con la chatID di ogni utente, inoltre se ogni utente ha associata una determinata chatID questa deve essere assegnata in fase di registrazione e deve essere salvata nel Catalog) (modifiche in rosso sul Catalog per implementare ciò)
* Vedere le statistiche relative alla propria stanza e di tutti i sensori di temperatura e umidità e i grafici su Thingspeak e gli ultimi aggiornamenti dei devices

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Daily\_monitor.py

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

È il service che deve fornire le statistiche sui sensori -> in ogni caso deve cercare nel Catalog ogni device a che field corrisponde e a che canale ThingSpeak, leggere il feed del canale, prendere solo il field di interesse e fare le statistiche su quei dati.

Fare molta attenzione perché se in un channel abbiamo 4 field ogni volta che invio anche solo i dati per uno dei field anche gli altri sono aggiornati ma Null. Fare anche attenzione che se mandi due dati contemporaneamente come per il sensore umidità e temperatura ne prende solo uno dei due ed è un pasticcio, si risolve sfasandoli di qualche secondo ma si incorre nel problema detto sopra.