

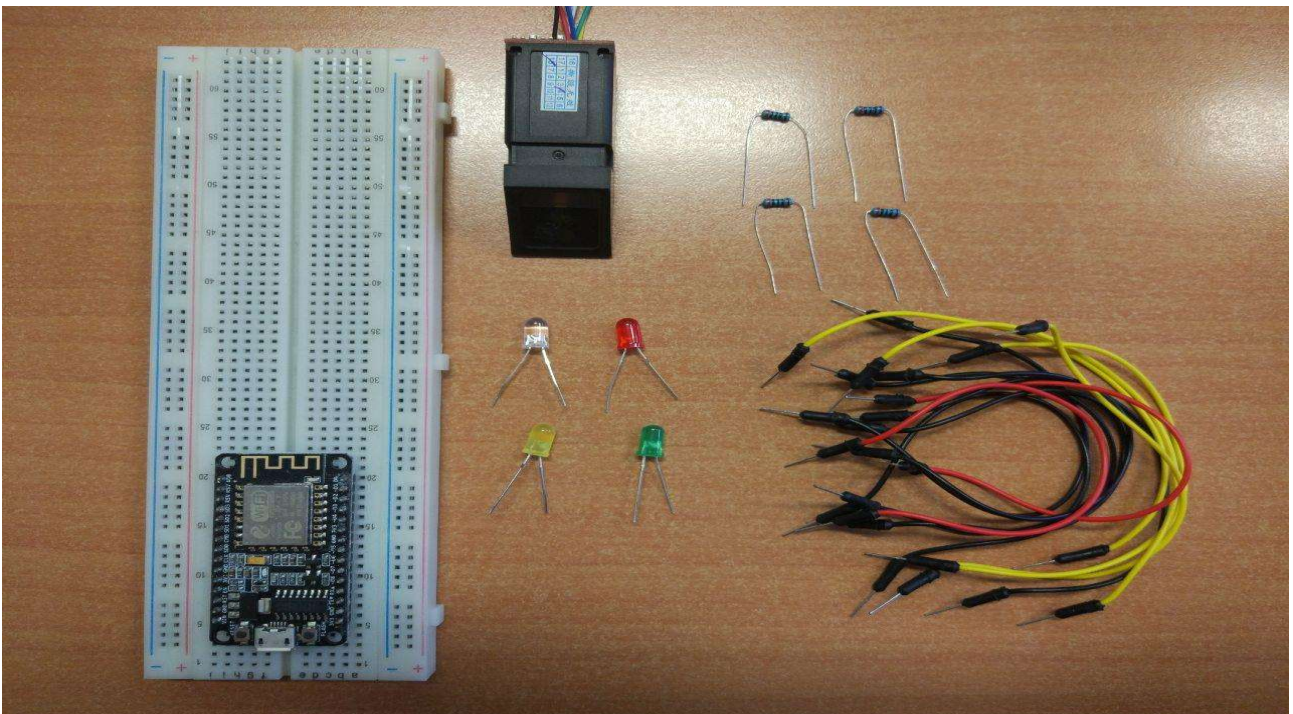
# Overview del Progetto

Il progetto ha come obiettivo l'implementazione di un sistema di autenticazione a due fattori, in questo caso, basato su una componente biometrica (il riconoscimento dell'impronta digitale) e da una parte di riconoscimento di un token inviato all'utente. Nella nostra implementazione, l'apertura della porta è stata simulata con l'accensione di un led verde.

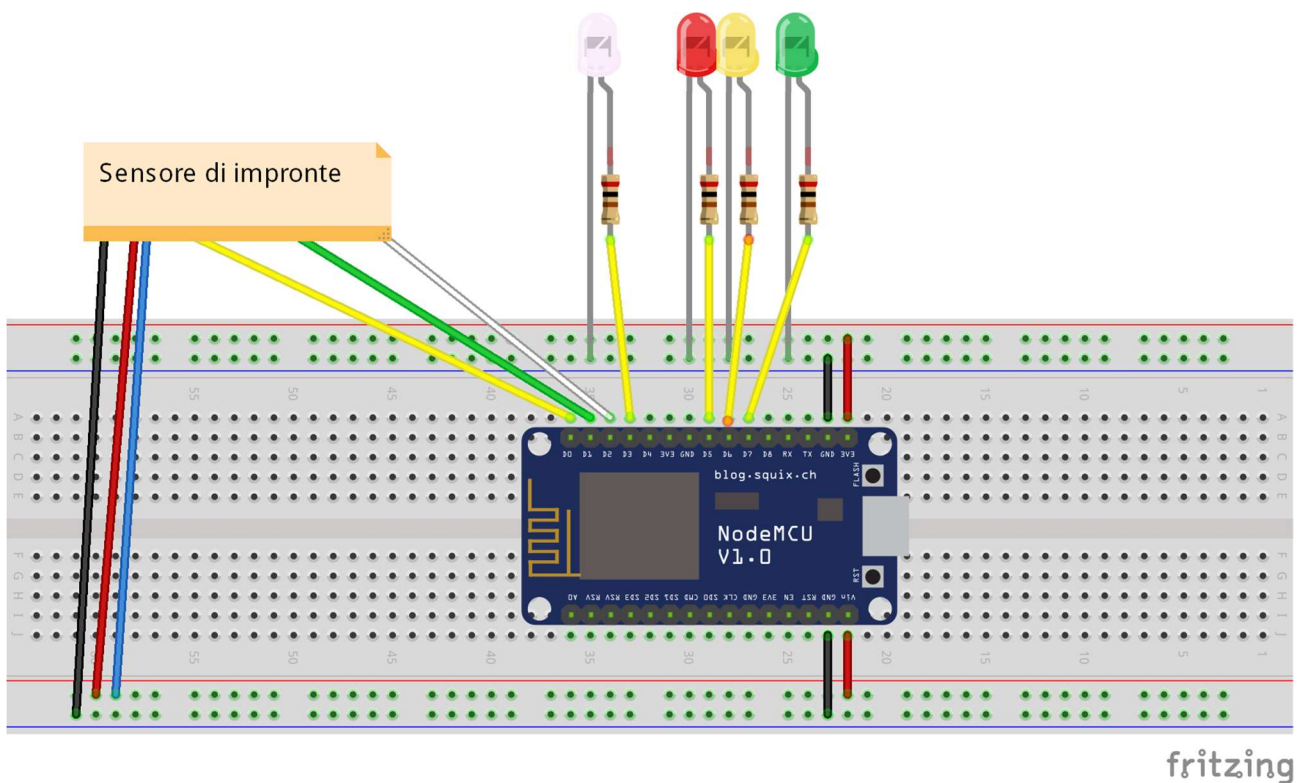
Per risparmiare potenza computazionale, abbiamo inoltre deciso di ricorrere ad un bot Telegram per l'invio del link per l'apertura della porta, in quanto questa piattaforma implementa nativamente un sistema di crittografia end-to-end, quindi questa fase è risultata del tutto trasparente.

## Strumenti Utilizzati

- Board NodeMcu 8266
- Lettore di impronte digitali ottico Kookye
- Led x4
- Resistenza 200  $\Omega$  x4
- Cavi



*Componenti utilizzati nel progetto 1*



*Schema dei collegamenti dei componenti 1*

## Servizi Utilizzati

- AWS RDS (Relational Database Server)
- AWS IoT Core
- AWS Lambda
- AWS API Gateway

## Relational Database Service

### Cosa è RDS?

È un servizio che consente di accedere alle funzionalità dei più comuni database relazionali quali : MySQL, SQL Server, PostgreSQL e così via. Permette quindi di creare e configurare il proprio database effettuando una copia di backup dello stesso e mantenendolo aggiornato all'ultima versione.

## Perché abbiamo utilizzato RDS?

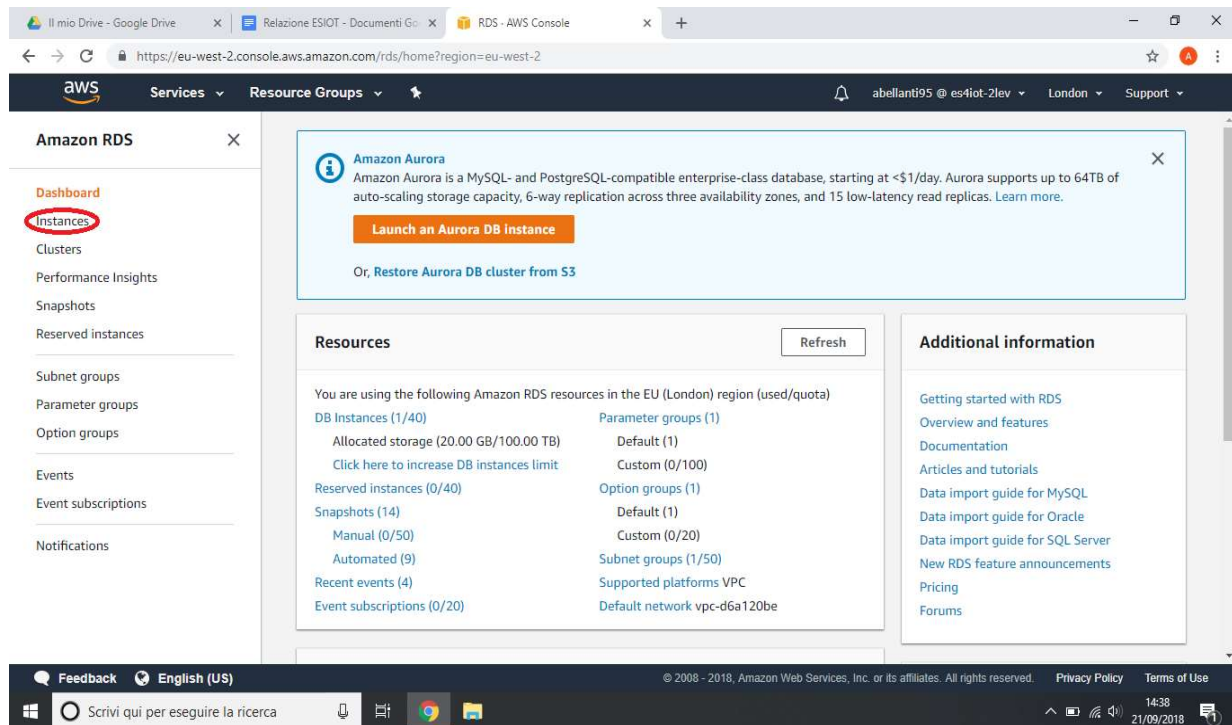
Il progetto, data in questo caso la sua natura puramente didattica, non richiedeva particolare attenzione a questa fase, in quanto dovendo salvare i dati relativi a un numero molto limitato di utenti e porte, sarebbe bastato anche un semplice file; ma per dare una soluzione da subito scalabile e sicura anche dal punto di vista della privacy, abbiamo deciso di gestire il tutto con un database relazionale composto da quattro tabelle. La scelta è ricaduta su questa famiglia di DBMS in quanto il tipo di dati che saranno salvati sono tutti molto simili tra loro, condizione ideale per questo tipo di database, e RDS ha permesso la creazione di un DB da subito senza la necessità di configurare i server.

Di seguito lo schema del database e la spiegazione dettagliata dei campi.

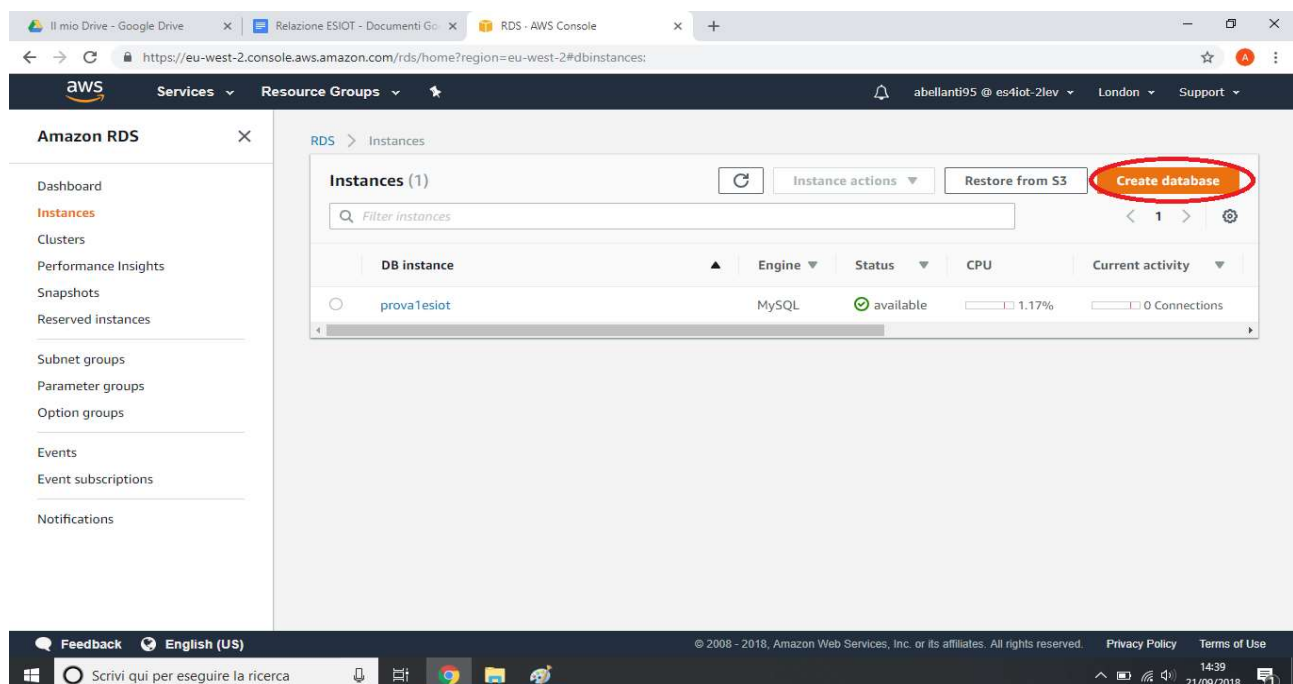
- 1) Users(id, nome, cognome, telegram\_id)
  - a) id: l'id associato al singolo utente. In questo caso, id è la chiave primaria
  - b) nome: il nome dell'utente
  - c) cognome: il cognome dell'utente
  - d) telegram\_id: l'id telegram dell'utente, utile per fare in modo che il bot invii il messaggio al giusto utente
- 2) door(id, building, floor)
  - a) id: l'id associato alla singola porta. In questo caso fa da chiave primaria
  - b) building: il nome dello stabile in cui è situata la porta
  - c) floor: il piano in cui è situata la porta
- 3) request(idPorta, idUtente, token, time, used)
  - a) idPorta: l'id della porta per cui è stata fatta quella richiesta.
  - b) idUtente: l'id dell'utente che ha effettuato quella richiesta
  - c) token: token generato casualmente
  - d) time: timestamp di quando la richiesta è stata effettuata
  - e) used: indica se il token è già stato utilizzato o meno
- 4) userdoor(idPorta, idUtente, idRiconosciuto)
  - a) idPorta: l'id della porta
  - b) idUtente: l'id reale dell'utente nel database
  - c) idRiconosciuto: l'id con cui quella porta riconosce quell'utente

## Come creare un'istanza RDS?

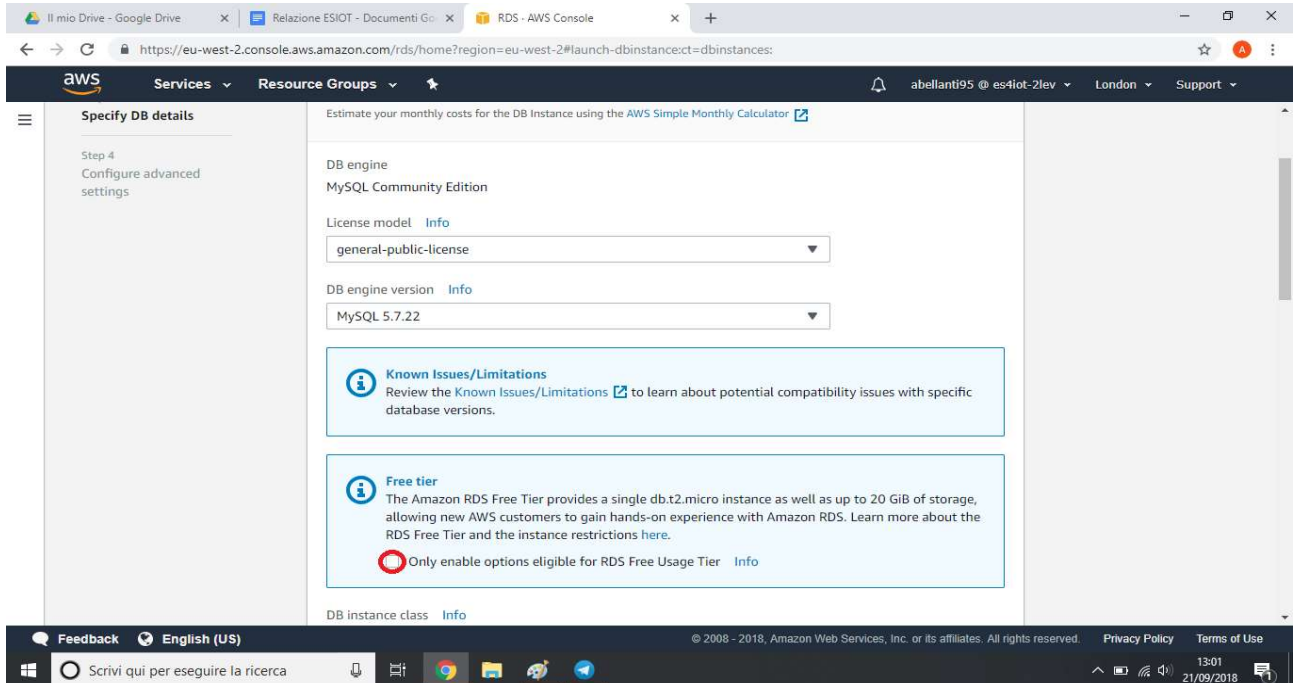
- accedere alla console AWS con le proprie credenziali
- accedere alla sezione RDS
- clickare su “instances”



- fare click su “Create database”



- Selezionare MySQL nella pagina che si aprirà
- Fare click su “next” in basso a destra
- Per rimanere nel free tier, selezionare “Dev/Test - MySQL” e clickare su next
- Selezionare l’engine MySQL preferito



- Per rimanere nel free tier, selezionare la checkbox indicata, andare a fondo pagina dare un nome all’istanza, inserire un master username e una master password per l’istanza e fare click su next
- Andare a fondo pagina e clickare su Create database
- Una volta lanciata l’istanza, usare un client MySQL per lanciare i comandi



# AWS IOT Core

## Cosa è AWS IOT Core?

AWS IOT core è un servizio messo a disposizione da AWS che permette di creare e gestire oggetti (things). Mette a disposizione, tra gli altri, dei servizi di shadowing e di trigger per le lambda.

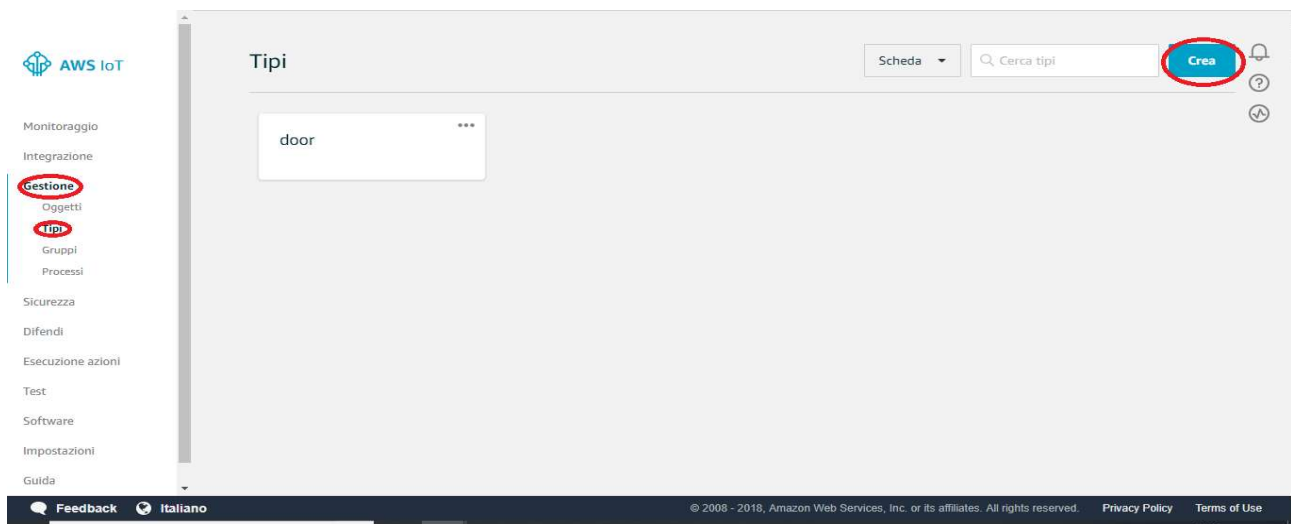
## Perché abbiamo usato AWS IOT Core?

Abbiamo deciso di utilizzare AWS IOT Core in quanto ci serviva un servizio di shadowing che ci permettesse allo stesso tempo di attivare una lambda in determinate condizioni.

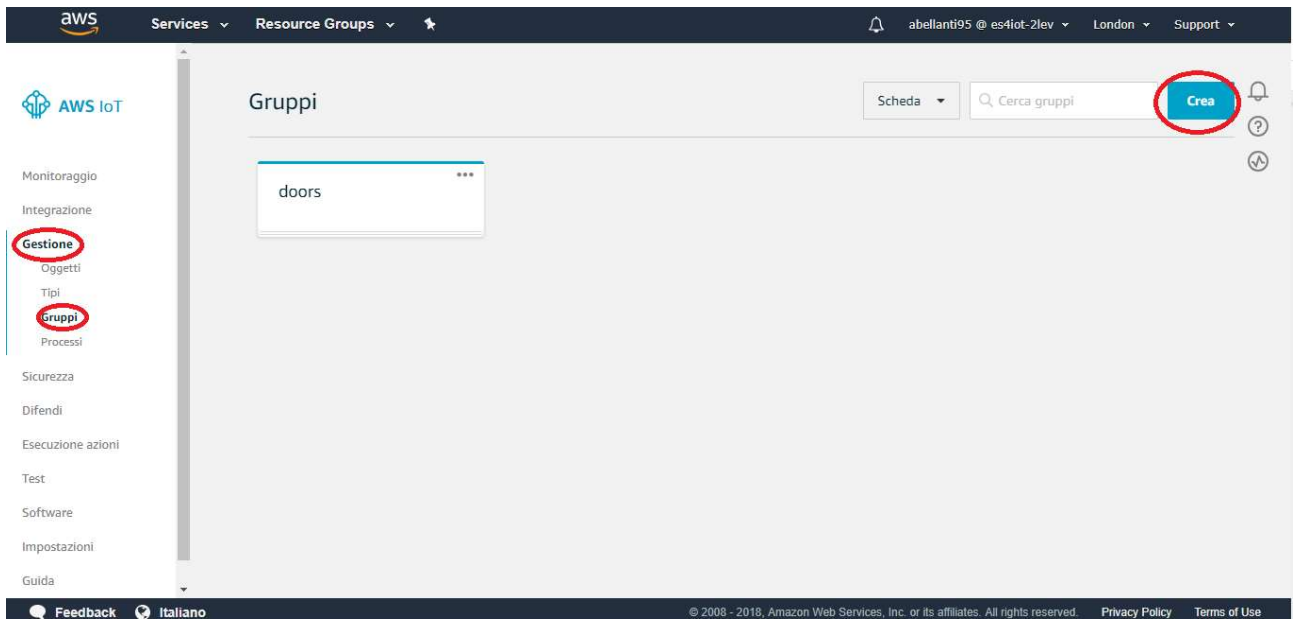
Questi sono due dei servizi che vengono offerti da AWS IOT Core, e quindi è stata la scelta che avrebbe permesso un miglior risultato finale, in quanto non facciamo affidamento su servizi di terze parti.

## Come usare AWS IOT Core

Per utilizzare questo servizio bisogna prima di tutto creare un tipo. Per farlo andare su “Gestione”, “Tipi” e successivamente “Crea”.



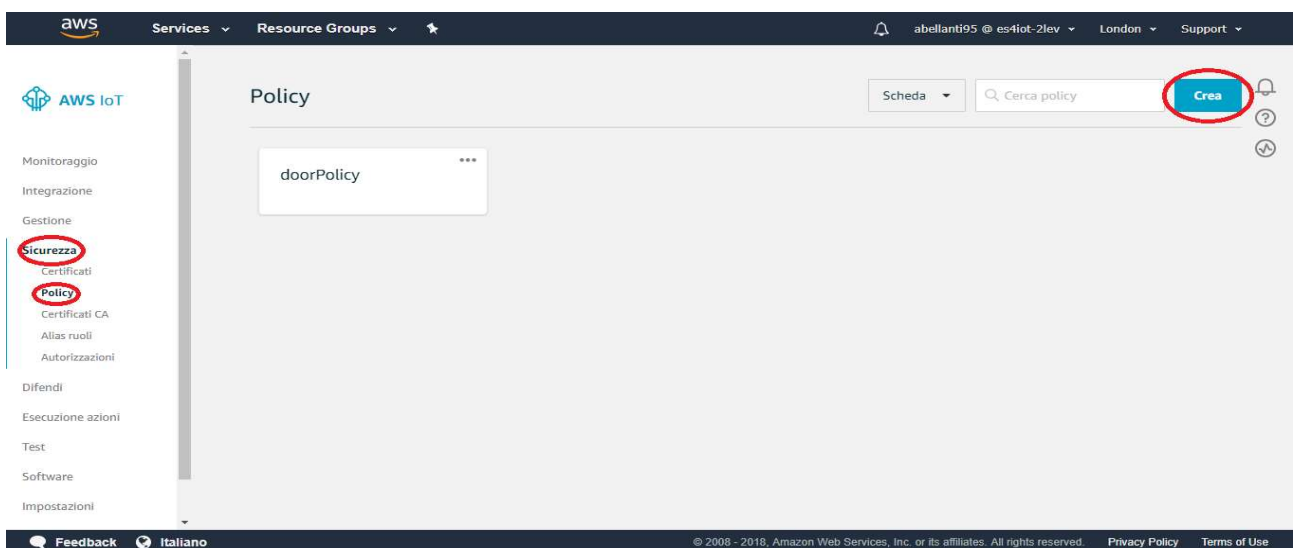
In “Nome” dare un nome al tipo e poi clickare su “Crea un tipo di oggetto” situato in basso a destra.



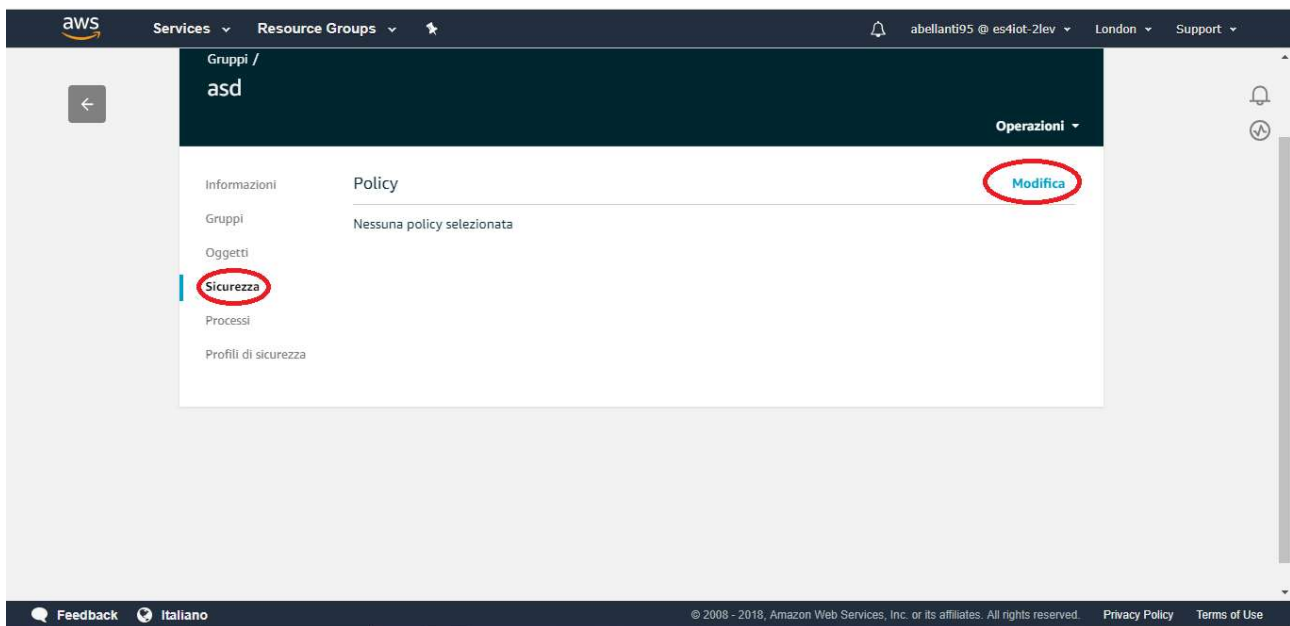
Il prossimo passo sarà creare un Gruppo. Clickare quindi su “Gestione”, successivamente su “Gruppi” e poi su “Crea”.

Anche nella prossima schermata dare un nome all’oggetto e successivamente fare click su “Crea gruppo di oggetti” situato in basso a destra.

Successivamente clickare su “Sicurezza” poi su “Policy” ed infine su “Crea”.



Dare quindi un nome alla nuova policy e successivamente clickare su “Modalità avanzata”.



Cancellare tutto il testo dalla casella di testo che si aprirà e sostituirlo con il seguente:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "lambda:*",
      "Resource": "*"
    }
  ]
}
```

Fare quindi click su “Crea” in basso a destra.

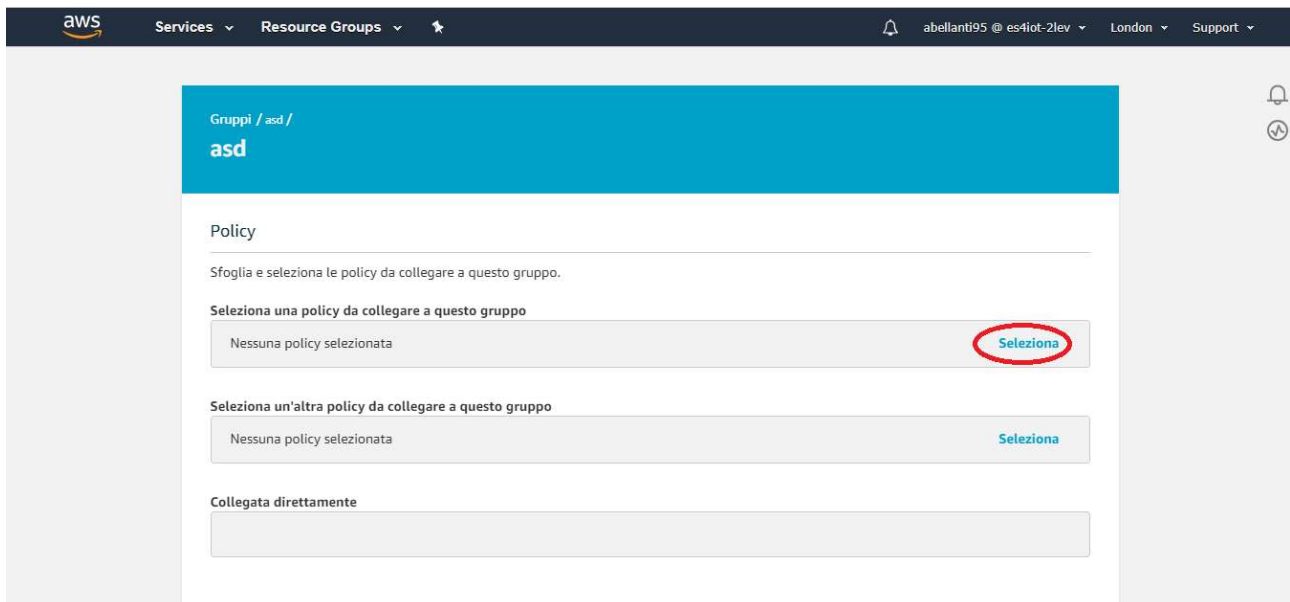
Ora che la policy è stata creata, tornare su “Gestione” -> “Gruppi”.

Cliccare sul gruppo precedentemente creato e spostarsi nella sezione “Sicurezza” e quindi fare click su “Modifica”

Nella schermata che si aprirà, clickare su Seleziona.



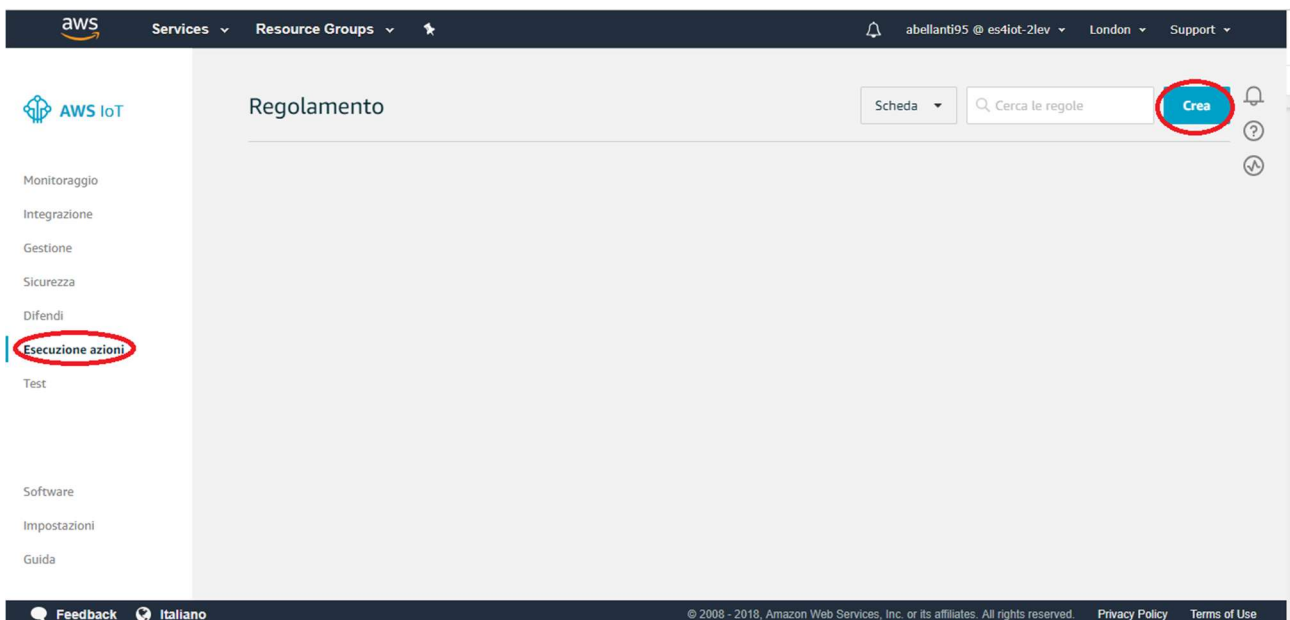




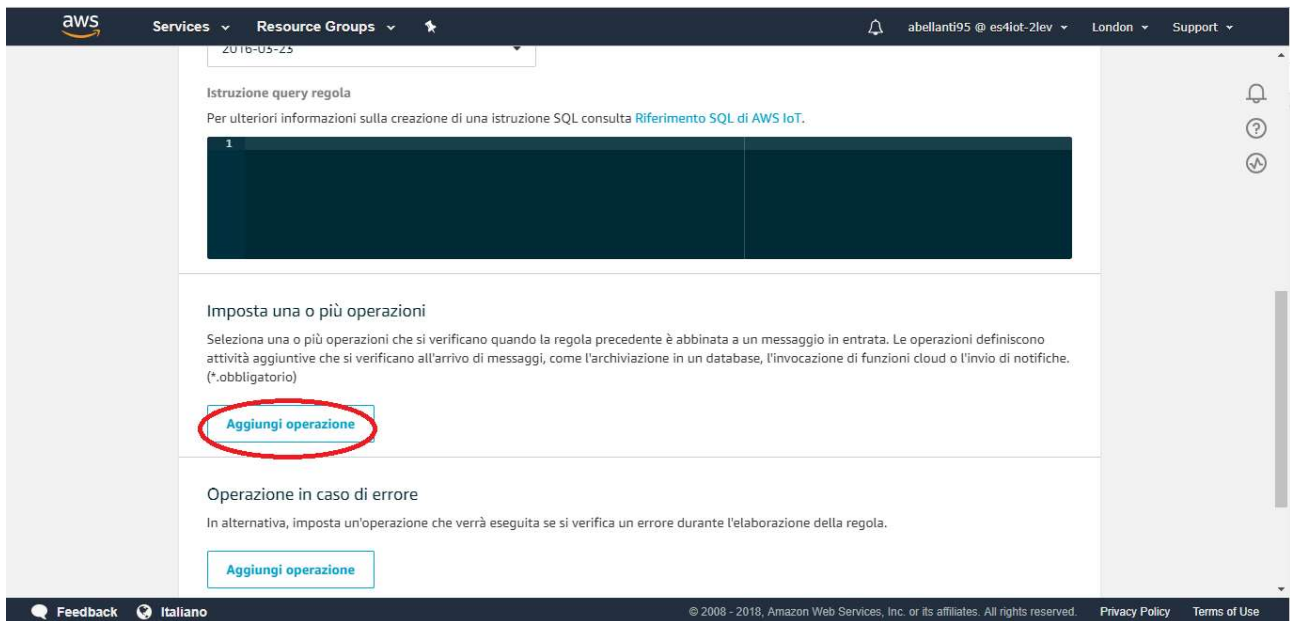
Selezionare quindi la policy appena creata e fare click su “Salva” in basso a destra. Ora clickare su Oggetti e quindi “Aggiungi un oggetto”.

Nella schermata che appare selezionare l'oggetto appena creato e fare click su “Aggiungi”.

Ora tornare alla schermata principale di AWS IOT Core e andare su “Esecuzione Azioni” E fare click su “Crea”.



Dare quindi un nome al trigger, impostare su “beta” la versione SQL e fare click su “Aggiungi operazione”



Nella schermata che si apre, selezionare “Invoca una funzione Lambda passando i dati dei messaggi” e fare click su “Configura operazione” in basso a destra.

Nella nuova schermata, fare click su “Seleziona” selezionare quindi la lambda che si occuperà di generare il token e inviare il messaggio, dopo di che, clickare su “Aggiungi operazione”.

Infine, nella casella di testo scrivere la seguente query SQL:

```
SELECT state.reported.id AS n, state.reported.idPorta as door FROM '$aws/things/+/shadow/update/accepted' WHERE state.reported.status="pending"
```

# Lambda

## Cosa è una Lambda?

Una funzione lambda è uno strumento che consente l'elaborazione serverless ossia la possibilità di creare ed eseguire applicazioni e servizi senza gestire alcun server. Una funzione lambda può essere scritta in: Java, Python o NodeJS; si tratta quindi di una porzione di codice che viene eseguito su AWS lambda in risposta a determinati eventi. È infatti possibile collegare la lambda con altri servizi AWS quali: AWS IoT, API Gateway, CloudWatch Logs, DynamoDB etc.

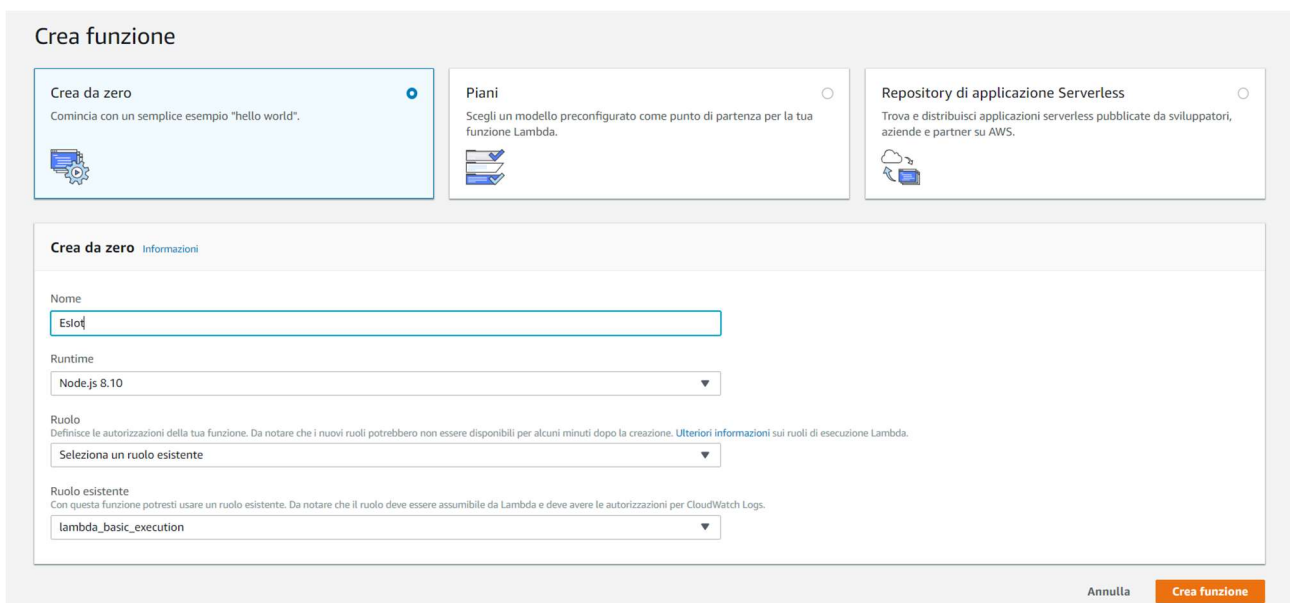
## Perché abbiamo utilizzato le lambda?

Avendo la necessità di gestire e manipolare dati su Cloud e non avendo a disposizione un server, è sembrata la soluzione migliore. AWS permette la creazione di un database sulla piattaforma senza la necessità di avere un server che lo 'ospiti' e le Lambda sono in grado di interrogare direttamente la base di dati e accedere alle informazioni necessarie all'elaborazione.

Grazie alla possibilità di integrare le richieste https e le API di telegram è stato possibile inoltre inviare il link necessario alla seconda fase dell'autenticazione.

## Come creare una lambda?

- Accedere ad AWS (Amazon Web Services) con le credenziali ...
- Selezionare dalla pagina AWS Services il servizio Lambda e su cliccare su GetStartedNow
- Cliccare su 'Crea funzione' e riempire i campi come mostrato in Fig 1.



**Crea funzione**

**Crea da zero** Informazioni

Comincia con un semplice esempio "hello world".

**Piani**

Scegli un modello preconfigurato come punto di partenza per la tua funzione Lambda.

**Repository di applicazione Serverless**

Trova e distribuisce applicazioni serverless pubblicate da sviluppatori, aziende e partner su AWS.

**Nome**

Eslof

**Runtime**

Node.js 8.10

**Ruolo**

Definisce le autorizzazioni della tua funzione. Da notare che i nuovi ruoli potrebbero non essere disponibili per alcuni minuti dopo la creazione. [Ulteriori informazioni](#) sui ruoli di esecuzione Lambda.

Seleziona un ruolo esistente

**Ruolo esistente**

Con questa funzione potresti usare un ruolo esistente. Da notare che il ruolo deve essere assumibile da Lambda e deve avere le autorizzazioni per CloudWatch Logs.

lambda\_basic\_execution

Annulla **Crea funzione**

Fig 1

- Dopo aver cliccato su Crea Funzione apparirà la schermata in Fig 2.

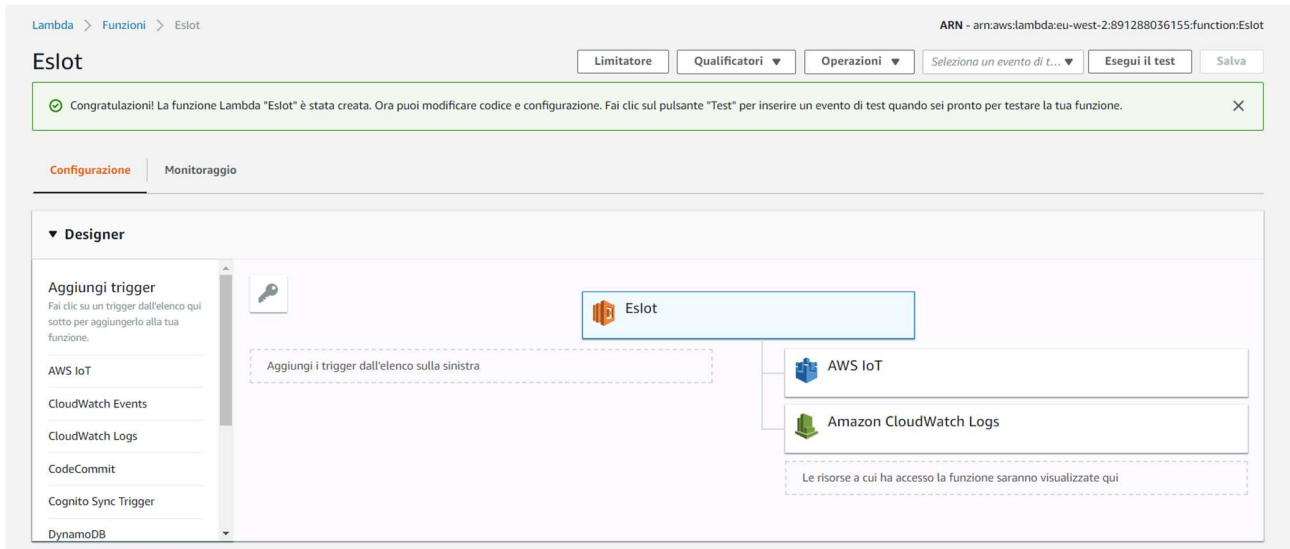


Fig 2

- A questo punto sarà necessario caricare il codice messo a disposizione cliccando su carica un file .ZIP come mostrato in fig 3.

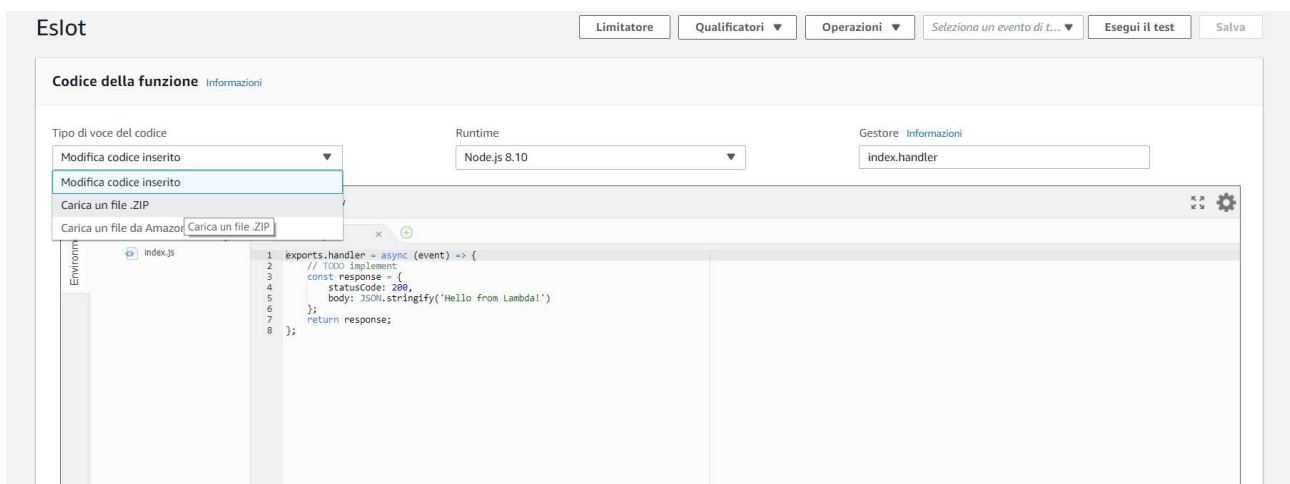


Fig 3

Eseguire questi passi sia per la Lambda Esiot sia per quella tokenStatus.

## Aggiunta dei Trigger e delle risorse alla Lambda : API Gateway

### Cosa è API Gateway?

È un servizio che permette di creare delle API che permettono alle applicazioni di accedere ai dati. È in grado di gestire centinaia di chiamate API simultanee e di gestire tutti i particolari legati ad autenticazione, monitoraggio e gestione delle API stesse.

### Perché abbiamo utilizzato API Gateway?

L'utilizzo di API Gateway si è reso necessario per triggerare (quindi eseguire) la Lambda relativa al controllo del Token. Il servizio è dotato di un endpoint API che se copiato nella barra di ricerca del Browser permette di eseguire la Lambda associata a quella API. Nel nostro caso, l'esecuzione della prima Lambda comporta l'invio di un messaggio con il link (l'endpoint dell'API) al bot telegram ed è l'apertura stessa del link da parte dell'utente a triggerare la seconda Lambda e permetterne l'esecuzione.

### Come collegare API Gateway alla Lambda?

Dopo aver caricato il codice nella Lambda, è necessario aggiungere un trigger in modo che la Lambda venga richiamata al verificarsi di determinati eventi.

Nel nostro specifico caso, dopo che l'utente ha ricevuto il link con il relativo token al bot telegram e lo ha aperto, ottenendo quindi l'accesso alla porta, la Lambda viene eseguita grazie alla ricezione da parte dell'endpoint del trigger (in questo caso API Gateway) di una richiesta Https.

1. Per aggiungere il trigger è necessario selezionarlo dalla lista di quelli disponibili all'interno del riquadro designer.

N.B. alla sinistra della Lambda vengono inseriti i Trigger, alla destra le risorse a che ha a disposizione.

2. Dopo aver aggiunto il trigger è necessario configurarlo, riempire i campi come mostrato in Fig 4 (la sicurezza deve essere impostata su Apri e non su AWS IAM) e cliccare su aggiungi.

**Configura trigger**

Configureremo un endpoint API Gateway con un tipo di integrazione proxy (ulteriori informazioni sul formato di input e di output). L'integrazione verrà attivata da tutti i metodi (GET, POST ecc.). Per configurare mappature del metodo o instradamenti del sottopercorso più avanzati, visita la [console Amazon API Gateway](#).

API  
Seleziona un'API esistente o creane una nuova.

Crea una nuova API

Sicurezza  
Configura il meccanismo di sicurezza dell'endpoint API.

AWS IAM

L'endpoint dell'API utilizzerà l'autenticazione IAM e richiederà all'intermediario le autorizzazioni IAM necessarie per richiamare l'API. [Ulteriori informazioni](#) sull'implementazione dell'autenticazione IAM.

▼ Altre impostazioni

Nome API  
Inserisci un nome per identificare l'API in modo univoco.

Eslot-API

Fase della distribuzione  
Il nome della fase di distribuzione dell'API.

default

Lambda aggiungerà le autorizzazioni richieste per Amazon API Gateway per chiamare la tua funzione Lambda da questo trigger. [Ulteriori informazioni](#) sul modello di autorizzazioni Lambda.

Annulla Aggiungi

Fig 4

3. Ritornare al menù principale e selezionare API Gateway dalla lista di servizi, in APIs apparirà l'API appena creata. Cliccare sul nome della stessa e poi su ANY. Dovrebbe apparire una schermata come quella mostrata in Fig 5.
4. Cliccare su Integration Request, togliere la spunta dall'opzione Use Lambda Proxy Integration e cliccare su ok su entrambe le finestre che appariranno.

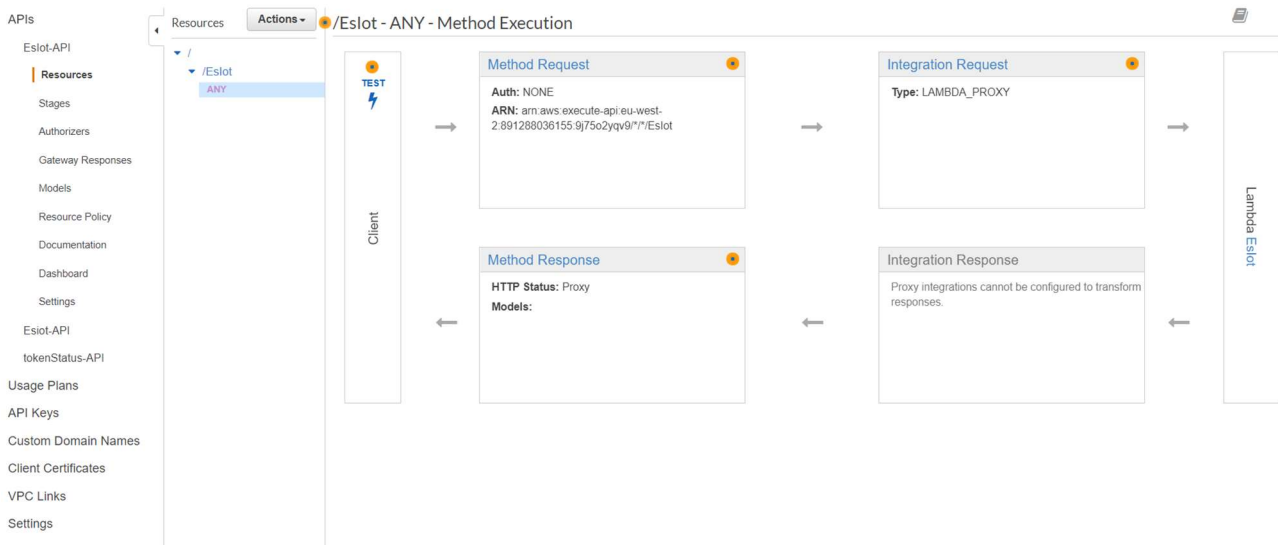
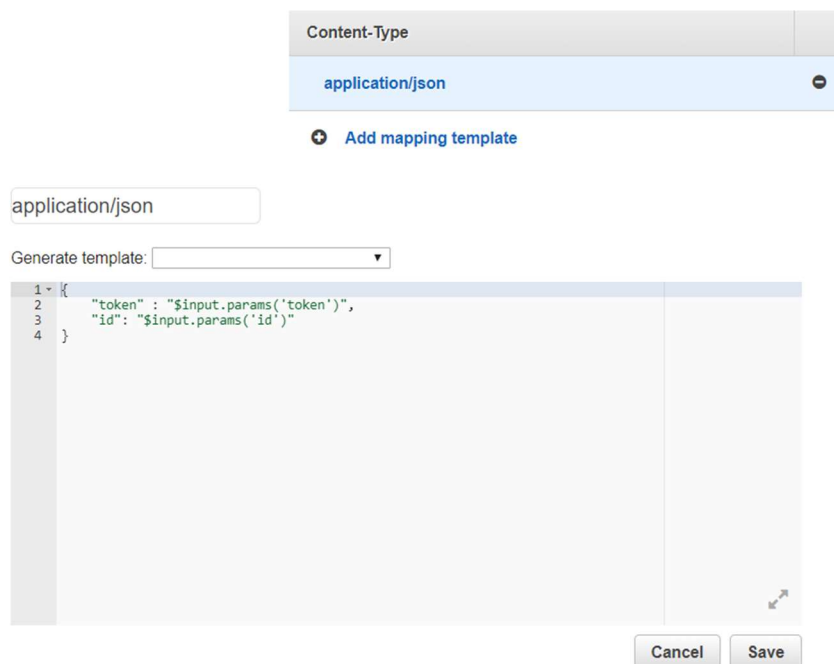


Fig 5

5. Sulla stessa pagina cliccare su Mapping Template e successivamente su Add mapping template e su Content-Type inserire la dicitura application/json. Cliccare su Yes nella finestra di richiesta di integrazione e inserire nel riquadro il codice come indicato in Fig 6 e poi cliccare su save.





*Fig 6*

6. Su Resources cliccare sul riquadro Method response e su Add Response Model all'interno di HTTP Status. In Content Type inserire text/html e in Models selezionare Empty.
7. Per rendere le modifiche permanenti, da Resource cliccare su Actions e poi su Deploy API. Nella finestra che appare selezionare default come Deployment Stage e cliccare su Deploy, infine cliccare su save changes.

## Difficoltà riscontrate durante lo svolgimento del progetto

La prima difficoltà con cui ci siamo dovuti scontrare è la totale mancanza di documentazione per quanto riguarda il sensore di impronte; l'unica fonte è stata un PDF reperito online che contiene delle spiegazioni parziali sulla libreria e un codice di esempio fornito con la libreria su cui è stato necessario eseguire del reverse-engineering per capire come collegare il sensore alla board, ma anche questo non è stato esaustivo in quanto non venivano usati tutti i cavi; il resto è stato scoperto a intuizioni e prove.

Un'altra grossa difficoltà è stata la documentazione non aggiornata per quanto riguarda AWS: è capitato praticamente sempre infatti che due fonti diverse descrivessero maniere del tutto differenti (e molte volte entrambe non corrette) per arrivare a uno stesso risultato.

## Come utilizzare il progetto

- 1) Eseguire l'enrollment dell'impronta digitale. Per farlo, flashare il codice fornito oltre al progetto
- 2) Aggiornare il database con la nuova impronta registrata (e in caso registrare il nuovo utente)
- 3) Utilizzare il progetto

## Possibili espansioni future

Il codice fornito è scritto in modo che alcune espansioni siano eseguibili in maniera molto semplice, per esempio aggiungendo un nuovo stato interno alla board (oltre ai già presenti "closed" "pending" e "open"), creando una nuova tabella nel database RDS (o modificando quelle esistenti) e scrivendo una lambda per un'autenticazione alternativa (prendendo quindi come spunto la lambda già esistente per l'autenticazione) è possibile nominare alcuni utenti "Amministratori" e quindi implementare una console utilizzando la seriale dedicata unicamente a questi utenti scrivendo semplicemente il codice per la gestione dei comandi.

Un'altra possibile espansione sarebbe quella di integrare il codice per l'enrollment dell'impronta digitale nel progetto principale, magari utilizzando un pulsante.

Infine, si potrebbe (per evitare di dover accedere sempre al DB con un client MySQL) realizzare una pagina web che contiene un form di registrazione e che esegua la query di aggiunta.

## Considerazioni finali sulla riuscita del progetto

Il progetto rispetta perfettamente le specifiche fornite, e lo fa con degli standard di sicurezza abbastanza elevati; infatti sia l'utilizzo di Telegram che quello di AWS danno una buona sicurezza sulla riservatezza. Per impedire in oltre che un eventuale avversario riesca a impadronirsi dei dati biometrici degli utenti, come ulteriore misura di sicurezza si è deciso di non inviare l'impronta per un riconoscimento lato server, ma di utilizzare una funzione di riconoscimento locale all'interno del sensore e inviare al server solamente un id.

### Link a GIT:

Schetch di riconoscimento dell'impronta e apertura della porta:

<https://github.com/Eagleheart95/EsiotSketchRiconoscimento>

Schetch di entrollment di una nuova impronta:

<https://github.com/Eagleheart95/Enroll>

Codice di generazione e invio del link con il token:

<https://github.com/Eagleheart95/Esiot>

Codice di apertura della porta:

<https://github.com/Eagleheart95/tokenStatus>