2022

# MyAcquarium

PROGETTAZIONE E IMPLEMENTAZIONE DI SISTEMI SOFTWARE IN RETE

MATTEO MORANDO, ANDREA FORNASIERO

UNIVERSITÀ DEL PIEMONTE ORIENTALE “A. AVOGADRO” | V.le Teresa Michel, 11, 15121 Alessandria AL

# Sommario

# Introduzione

MyAcquarium è un sistema per gestire autonomamente dispositivi adoperati in acquari di tipo tropicale e marino. L'applicazione web concede all'utente di registrarsi per il proprio spazio personale dove creare gli acquari secondo i dispositivi in possesso e modificare le regole di gestione a proprio piacimento. Il monitoraggio permette di visualizzare i valori dell'acquario con grafici delle serie storiche.

MyAcquarium riduce il tempo necessario alla manutenzione dei propri acquari offrendo un sistema completamente autonomo e personalizzabile dall'utente. Nessuna preoccupazione nei giorni di assenza e ferie grazie al controllo a distanza dei dispositivi per una rapida occhiata ai valori dell'acquario. Il cibo scarseggia o un dispositivo ha problemi? Anche a questo ci pensa MyAcquarium che, grazie al sistema di avvisi, permette di sapere in tempo reale quando un dispositivo non funziona correttamente.

Il sistema è adatto a neofiti che vogliono iniziare il loro percorso, o a professionisti che preferiscono un approccio automatizzato. In fase di creazione è possibile visualizzare le varie tipologie di pesci adatte al tipo di acquario che si vuole realizzare, e vengono impostate automaticamente le regole di controllo migliori, che saranno sempre visibili e modificabili dall'utente.

# Specifiche funzionali

## Casi d'uso

Il sistema è usufruibile dall'utente solo previa registrazione. Dopo aver effettuato l'accesso con username e password viene caricata la pagina dell'utente con i propri dati. Cliccando sui pulsanti presenti in questa pagina, o navigando mediante la barra di navigazione, è possibile creare un nuovo acquario o visualizzare gli acquari già presenti ed eventuali loro avvisi. Queste ultime due funzioni non sono utilizzabili nella pagina del profilo utente se nessun acquario è disponibile, o portano ad un avviso di creazione di un primo acquario se selezionate dalla barra di navigazione.

Nella pagina di creazione, l'utente sceglie un nome e le tipologie di pesci associate per riconoscere l'acquario, e aggiunge i vari dispositivi in suo possesso selezionando il loro codice modello. Il distributore di cibo, il sistema di illuminazione, e la pompa, sono necessari in ogni acquario, mentre i tester, i vari regolatori, e i filtri, sono facoltativi a seconda dei pesci che abiteranno in quell'acquario. Se si ha difficoltà nella scelta dei pesci, è possibile cliccare su un collegamento per conoscere più in dettaglio le tipologie adatte al tipo di acquario considerato.

Conclusa la creazione dell'acquario l'utente arriva alla pagina degli acquari, dove visualizza tutti gli acquari a disposizione, avendo il permesso di controllare i dettagli, modificare i dispositivi e le regole di gestione, eliminarli, o crearne di nuovi.

Nella pagina di informazioni sullo specifico acquario si visualizzano indicazioni generali circa modello e data di installazione dei dispositivi presenti, stato e valori attuali ricevuti, regole di gestione, e grafici delle serie storiche.

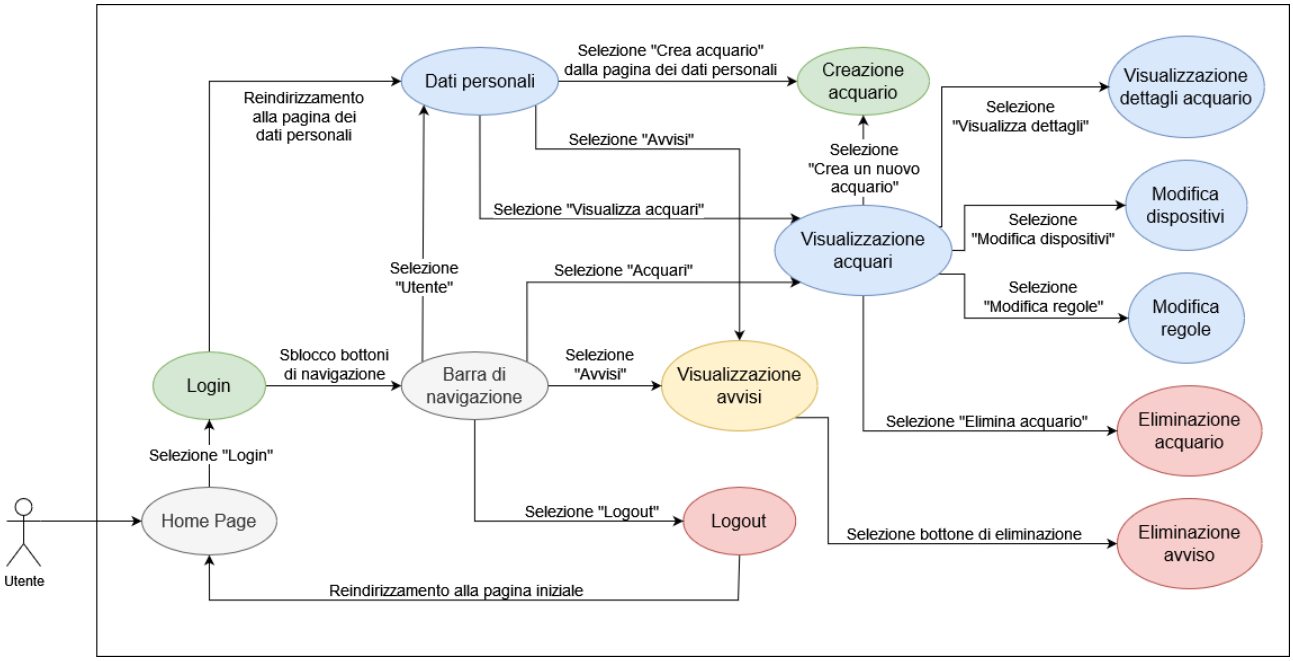
Nella pagina di modifica dei dispositivi l'utente visualizza un modulo molto simile a quello di creazione dell'acquario, dove gli sarà possibile variare nome, tipologie di pesci, e dispositivi dell'acquario. Non è possibile alterare il tipo di acquario: se viene deciso un cambiamento drastico che porta al cambio di tipo è necessario eliminare l'acquario e ripartire da zero con un nuovo acquario del tipo variato.

Nella pagina di modifica delle regole di gestione è possibile variare l'amministrazione automatica dell'acquario scegliendo parametri personali. Se si è inesperti non è indicato modificare la gestione dei valori dell'acquario prestabilita dal sistema.

Nella pagina degli avvisi si seleziona un acquario tra quelli realizzati e vengono visualizzati gli errori inviati da dispositivi guasti o con problemi dell'acquario selezionato. Si raccomanda l'utente di prendere nota degli avvisi prima di procedere alla loro eliminazione.

Il pulsante rosso con la scritta "Logout" presente nella barra di navigazione consente all'utente di disconnettersi dal sistema, ritornando alla pagina principale con la visualizzazione del pulsante verde con la scritta "Login" di accesso.

Il grafico dei casi d'uso sotto presentato mostra i passi che l'utente deve eseguire per arrivare a una determinata pagina o compiere l'azione desiderata. I colori rappresentano i veri colori dei pulsanti per giungere a quel caso d'uso: i pulsanti di login e creazione sono in verde, in giallo i pulsanti per gli avvisi, in rosso i pulsanti più pericolosi di eliminazione e logout, in blu tutti gli altri pulsanti. L'utente viene invitato a confermare l'eliminazione di un acquario o di un avviso per evitare la rimozione accidentale.




## Home Page

La Home Page è la pagina iniziale dell'applicazione web. Da qui è possibile effettuare l'autenticazione o navigare nelle varie pagine disponibili una volta autenticati grazie alla barra di navigazione. con bottoni che vengono visualizzati o meno a seconda dell'autenticazione dell'utente. È possibile ritornare alla Home Page in qualsiasi momento cliccando su "MyAcquarium".

## Login

Premendo sul pulsante di Login l'utente viene reindirizzato al servizio di autorizzazione (Keycloak), dove potrà autenticarsi inserendo il suo username e la sua password, o registrarsi inserendo i suoi dati personali. Se si viene autorizzati correttamente, si ottiene un token per accedere ai servizi interni dell'applicazione. Le

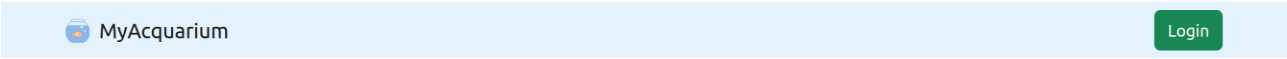
richieste successive verranno automaticamente autenticate fino alla scadenza del token, dove l'utente verrà invitato a rieseguire l'autenticazione, o fino alla disconnessione mediante il tasto di Logout.

### Logout

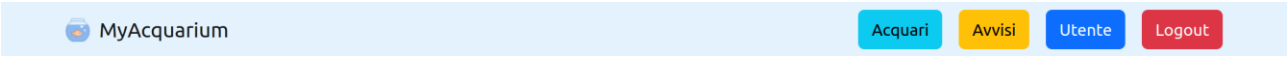
Il pulsante di Logout permette all'utente di disconnettersi dall'applicazione, ritornando alla pagina principale. È necessario effettuare una nuova autenticazione per accedere ai servizi.

### Barra di navigazione

La barra di navigazione è sempre visibile nel lato superiore di ogni pagina web dell'applicazione. Scegliendo i pulsanti opportuni, che cambiano a seconda dell'autenticazione dell'utente, è possibile autenticarsi, disconnettersi, e raggiungere la Home Page o le pagine di visualizzazione degli acquari, degli avvisi, e dei dati personali.




*Barra di navigazione, utente non autenticato*




*Barra di navigazione, utente autenticato*

### Pagina dati utente

In questa pagina, l'utente visualizza i suoi dati personali inseriti in fase di registrazione: username, nome, cognome, ed email. Sono presenti pulsanti per la creazione e la visualizzazione degli acquari e dei loro avvisi. I pulsanti di visualizzazione sono disattivati segnalati dalla comparsa di un avviso nel caso non si abbia nessun acquario a disposizione.

### Creazione acquario

Qui viene presentato un modulo per inserire i dati necessari alla creazione di un nuovo acquario. Il nome, il tipo di acquario, i codici del distributore di cibo, dell'illuminazione, e della pompa, sono obbligatori, e non è possibile proseguire se non si vengono selezionati. È possibile creare acquari con lo stesso nome, anche se sconsigliato per facilitare il riconoscimento tra di essi. Il campo "Tipologie di pesci" non è obbligatorio se si è indecisi: il riferimento di informazione per conoscere le tipologie di pesci adatte al tipo di acquario presente può tornare utile per farsi un'idea più precisa. I tester sono selezionabili mettendo una spunta vicino al nome del tester a disposizione. I regolatori disponibili solo in presenza del tester specificato potranno essere selezionati solo dopo aver introdotto il tester dichiarato. Una volta terminato l'inserimento dei dati, cliccando su "Crea acquario" è possibile confermare i dati e venire reindirizzati alla pagina degli acquari, dove sarà possibile visualizzare l'acquario appena creato. Il pulsante Annulla riporta l'utente alla pagina degli acquari senza nulla di fatto.

## Visualizzazione acquari, dettagli, e avvisi

La pagina di visualizzazione degli acquari mostra gli acquari creati dall'utente, permettendo di visualizzare i dettagli di ogni singolo acquario, modificare i dispositivi o le regole di gestione, ed eliminarlo. In questa pagina sono presenti solo il nome e il tipo di acquario.

Cliccando su "Visualizza acquari" l'utente può prendere nota di tutte le caratteristiche dell'acquario. Nel titolo della pagina viene indicato ancora una volta il nome e il tipo di acquario, nel sottotitolo sono visibili le tipologie di pesci.

Un messaggio di attenzione raccomanda di attendere qualche minuto se i dati non sono disponibili: i dispositivi inviano i dati ogni minuto, quindi se l'acquario è stato appena creato, lasciare attendere almeno 1 minuto prima di visualizzare di nuovo i dati. Il sistema di monitoraggio invia i dati ogni 5 minuti: anche in questo caso i grafici delle serie storiche si aggiorneranno all'incirca ogni 5 minuti. Ovviamente non è possibile monitorare valori di cui non si hanno i tester, quindi il grafico rimarrà vuoto. Lo stato non ancora disponibile risulterà con la scritta "non ancora disponibile", mentre i valori avranno la scritta "N/A".

Premendo sui pulsanti nella pagina è possibile visualizzare: le informazioni dei dispositivi generali (tipologia di dispositivo, modello, data di installazione), lo stato e i valori attuali (tipologia di dispositivo, stato, valori), le regole di gestione (tipologia di dispositivo o valore acquario, regole di gestione), e i grafici.

È possibile visualizzare gli avvisi dalla pagina dei dati personali o dalla barra di navigazione cliccando su "Avvisi". Selezionare l'acquario di cui si vogliono visualizzare gli errori, prendere nota di eventuali errori, ed eliminarli cliccando sull'apposito pulsante.

## Modifica dispositivi

In caso di modifiche all'acquario, l'utente può modificare i dispositivi selezionando i nuovi. È possibile cambiare il nome e le tipologie di pesci nell'acquario in caso di aggiunta di altri pesci per un più facile riconoscimento. Il tipo di acquario non è modificabile. Per aggiungere un nuovo dispositivo è sufficiente selezionare il suo codice dal menù a tendina. La scritta "Invariato" indica che non verranno applicate modifiche, mentre "Nessuno" indica che il dispositivo in questione è stato rimosso. Ovviamente non è possibile rimuovere il distributore di cibo, l'illuminazione, e la pompa. Fare click su "Modifica acquario" per confermare la nuova selezione, o su "Annulla" per annullare e ritornare alla pagina degli acquari.

## Modifica regole

In questa pagina, il titolo indica nuovamente il nome e la tipologia di acquario, per evitare di modificare regole di un altro acquario. Le regole sono modificabili scrivendo o selezionando un orario, o spostando un cursore a sinistra (valore diminuito) o a destra (valore incrementato).

L'orario primo pasto è modificabile tra 00:00 e 11:59, mentre quello secondo pasto tra 12:00 e 23:59.

Stesso meccanismo per l'inizio luce diurna, che sarà modificabile tra 00:00 e 11:59, e inizio luce notturna, con un raggio variabile tra 12:00 e 23:59.

La temperatura è modificabile tra 15 °C di minima e 35 °C di massima.

La percentuale di acqua è modificabile tra 90% di minima e 99% di massima.

Il valore di anidride carbonica è modificabile tra 10 mg/l di minimo e 60 mg/l di massimo.

Il valore di nitrati è modificabile tra 0 mg/l di minimo e 100 mg/l di massimo.

Il valore di fosfati è modificabile tra 0 mg/l di minimo e 0.5 mg/l di massimo.

Il valore di silicati è modificabile tra 0 mg/l di minimo e 1 mg/l di massimo.

### Eliminazione acquari e avvisi

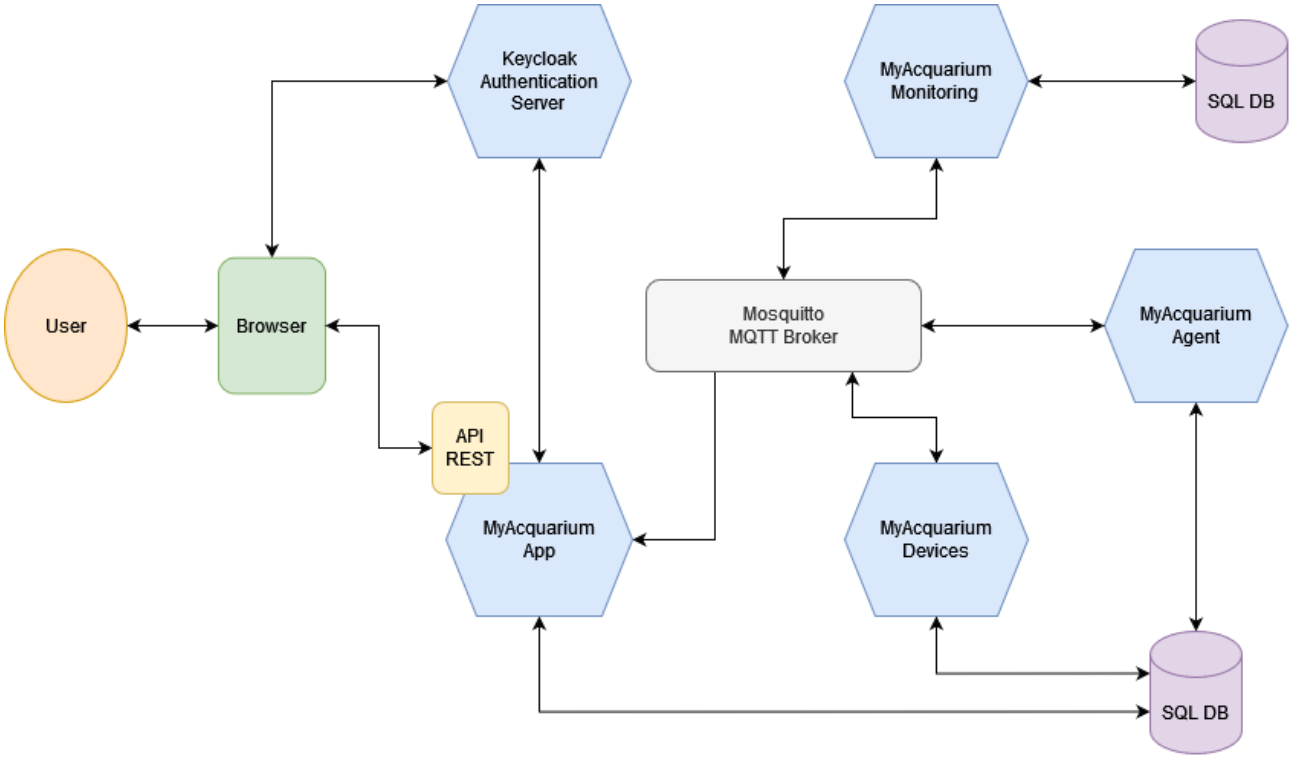
L'eliminazione di un acquario avviene tramite la pressione del pulsante "Elimina acquario" vicino all'acquario da eliminare. La richiesta chiede una doppia conferma per assicurarsi di voler eliminare l'acquario ed è irreversibile, quindi una volta eliminato non è possibile tornare indietro.

Anche l'eliminazione di un avviso richiede una conferma aggiuntiva per assicurarsi di averne preso nota prima di eliminarlo per sempre.

## Architettura del software

L'architettura del software è organizzata in 4 microservizi indipendenti che comunicano tra loro tramite API ben definite denominati MyAcquariumApp, MyAcquariumAgent, MyAcquariumDevices, e MyAcquariumMonitoring. Questi microservizi fanno uso di Spring Boot: sono costruiti sul framework Spring basato su piattaforma Java e sono autonomi e pronti per ambienti di produzione. La versione di Spring Boot utilizzata per ogni microservizio è la 2.7.5 con Java 19.

La compilazione dell'applicazione e le dipendenze del progetto sono gestite con Apache Maven 3.8.6. Ogni microservizio contiene un file "pom.xml", basato sul concetto di Project Object Model, dove sono presenti le versioni di librerie necessarie alla compilazione. Il plug-in Spring Boot Maven permette inoltre di impacchettare i microservizi in file eseguibili in formato jar (Java Archive), così da avere un semplice file con tutto il necessario per l'avvio.




MyAcquariumApp è l'applicazione web, usufruibile dall'utente via web mediante l'interfaccia API REST.

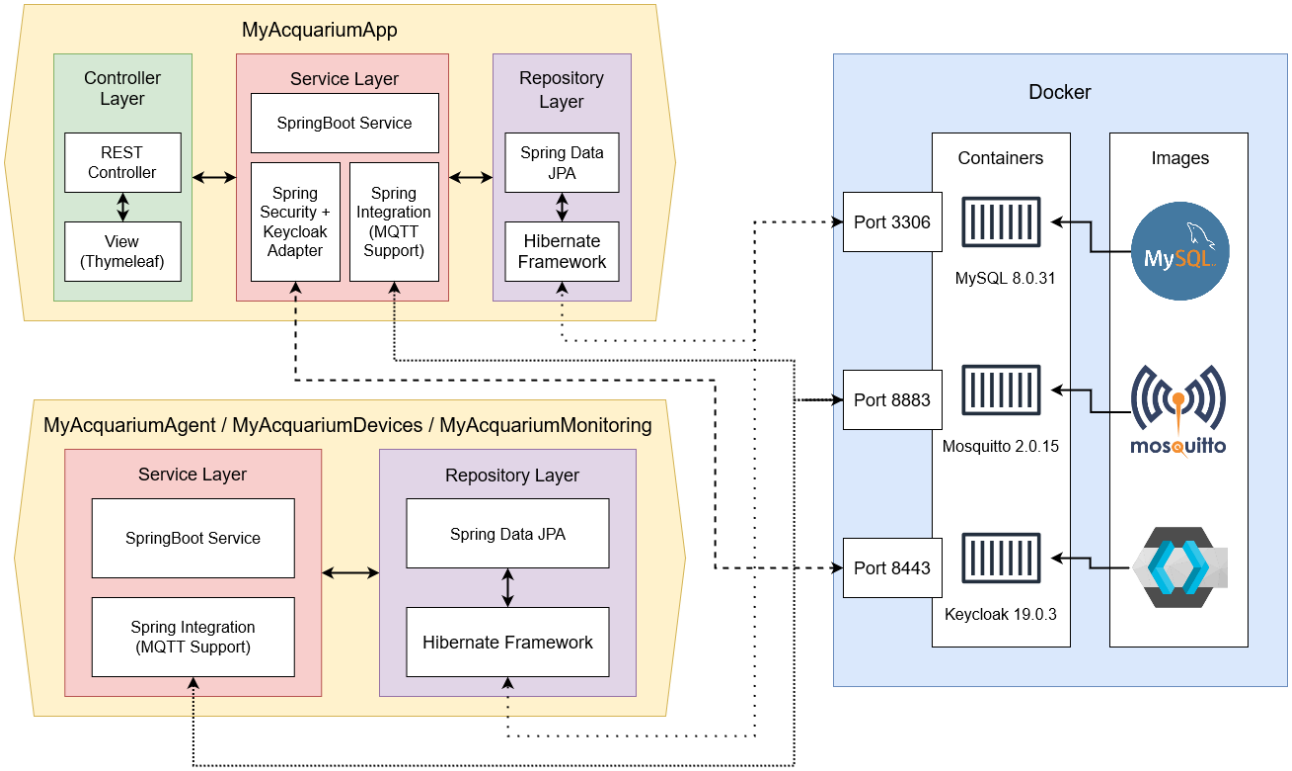
Keycloak è il server di autenticazione, il quale comunica con l'App e il browser dell'utente per autenticarlo tramite protocollo OpenID Connect (OIDC).

MyAcquariumAgent è l'agente che gestisce in autonomia i dispositivi secondo le regole di gestione dichiarate.

MyAcquariumDevices è il microservizio che simula i dispositivi, inviando valori fittizi e ricevendo regole di gestione da MyAcquariumAgent.

MyAcquariumMonitoring è il sistema di monitoraggio: salva sul database i valori dell'acquario ricevuti dall'agente che sono stati inviati dai dispositivi, e invia alla WebApp una lista di valori storici.

I microservizi App, Devices, e Agent, condividono lo stesso database; Monitoring ha un database separato.




I microservizi MyAcquariumAgent, MyAcquariumDevices, e MyAcquariumMonitoring (nell'immagine riuniti per semplicità) sono suddivisi in 2 layer:

- Service Layer, dove sono presenti i servizi di SpringBoot necessari ad eseguire l'applicazione Spring, e il supporto al protocollo Message Queuing Telemetry Transport (MQTT), che comunica col message broker Mosquitto mediante la porta 8883;
- Repository Layer, che si occupa della persistenza dei dati nel database grazie al framework Java Persistence API (JPA) e Hibernate, che fornisce un servizio di Object-relational mapping (ORM) permettendo di mappare oggetti Java su un database relazionale come MySQL, al quale comunica mediante la porta 3306.

Il microservizio MyAcquariumApp possiede i 2 layer elencati precedentemente, in aggiunta al Controller Layer, che permette l'implementazione di un'interfaccia API REST.

Il Controller comunica con il motore di template Thymeleaf per la creazione delle pagine HTML visualizzate dall'utente. Thymeleaf fa uso del framework Bootstrap 5.2.2 per il design dei template, della libreria JavaScript jQuery 3.6.1, e della libreria JavaScript Chart.js 3.9.1 per i grafici.

Nel Service Layer è inoltre presente il supporto a Spring Security e Keycloak per autorizzare l'utente ad accedere alle risorse offerte dal servizio. Il microservizio comunica col server Keycloak mediante la porta 8443.

L'utente può comunicare con la WebApp di MyAcquarium attraverso il protocollo HTTPS sulla porta 8444. L'autenticazione col server avviene utilizzando il Mutual TLS (mTLS). In questo modo non è solo il client a verificare il server, ma anche il server controllerà che il client sia chi afferma di essere verificando se dispone della chiave privata corretta. Il browser dell'utente (in questo caso il client) deve predisporre del certificato dell'utente e mostrarlo alla WebApp se vuole ottenere l'accesso al sito. Il certificato deve risultare valido ed emesso da una Certification Authority (CA) fidata. MyAcquariumApp predispone anch'essa di un certificato da mostrare all'utente, e avrà inoltre una lista di CA ritenute fidate per poter validare i certificati dei client rilasciate dalle sole CA presenti nella lista. In fase di autenticazione l'utente comunica con Keycloak, e anche in questo caso verrà mostrato il certificato per l'autenticazione reciproca.

Il database MySQL, il message broker Mosquitto, e il server Keycloak, sono container Docker: unità autonome con tutto il necessario per eseguire l'applicazione specificata dall'immagine.

### Container Docker: MySQL, Keycloak, Mosquitto

```
docker run \
  --name mysql \
  -p 3306:3306 \
  -e MYSQL_ROOT_PASSWORD=root \
  -d mysql:8.0.31
```


```
docker run \
  --name keycloak \
  -p 8443:8443 \
  -e KEYCLOAK_ADMIN=admin \
  -e KEYCLOAK_ADMIN_PASSWORD=admin \
  -e KC_HTTPS_KEY_STORE_FILE=/opt/keycloak/conf/keycloak-keystore.jks \
  -e KC_HTTPS_KEY_STORE_PASSWORD=password \
  -e KC_HTTPS_TRUST_STORE_FILE=/opt/keycloak/conf/my-truststore.jks \
  -e KC_HTTPS_TRUST_STORE_PASSWORD=password \
  -e KC_HTTPS_CLIENT_AUTH=required \
  -v $PWD/keycloak-keystore.jks:/opt/keycloak/conf/keycloak-keystore.jks \
  -v $PWD/my-truststore.jks:/opt/keycloak/conf/my-truststore.jks \
  -v $PWD/realm.json:/opt/keycloak/data/import/realm.json \
  quay.io/keycloak/keycloak:19.0.3 \
  start-dev --import-realm
```

```
docker run -it \
  --name mosquitto \
  -p 8883:8883 \
  -v $PWD/mosquitto.conf:/mosquitto/config/mosquitto.conf \
  -v /mosquitto/data \
  -v /mosquitto/log \
  -v $PWD/ca.crt:/mosquitto/ca_certificates/ca.crt \
  -v $PWD/server.crt:/mosquitto/certs/server.crt \
  -v $PWD/server.key:/mosquitto/certs/server.key \
  -v $PWD/acl_file.txt:/mosquitto/acl/acl_file.txt \
  eclipse-mosquitto:2.0.15
```


Nell'immagine qui sopra sono elencati i comandi per la creazione dei container dei servizi necessari.

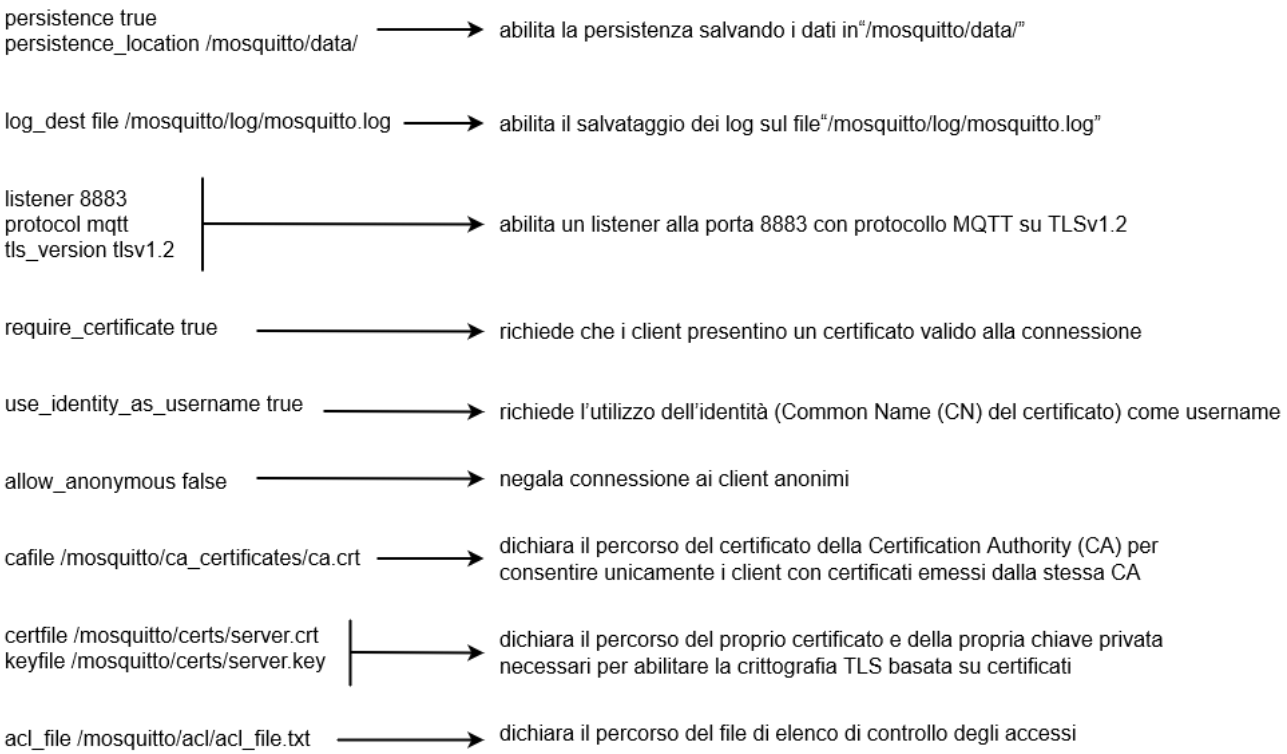
Il primo comando genera un container di nome “mysql” sull’immagine “mysql” di versione “8.0.31”. Con “-p 3306:3306” si lega la porta 3306 del container alla porta 3306 dell’host per comunicare attraverso quel numero di porta. Il parametro “-e MYSQL_ROOT_PASSWORD=root” imposta a “root” la password dell’utente “root”.

Il secondo comando genera un container di nome “keycloak” sull’immagine “quay.io/keycloak/keycloak” di versione “19.0.3”. Viene aperta la porta 8443 per le comunicazioni e viene scelto “admin” come username e password di amministrazione.

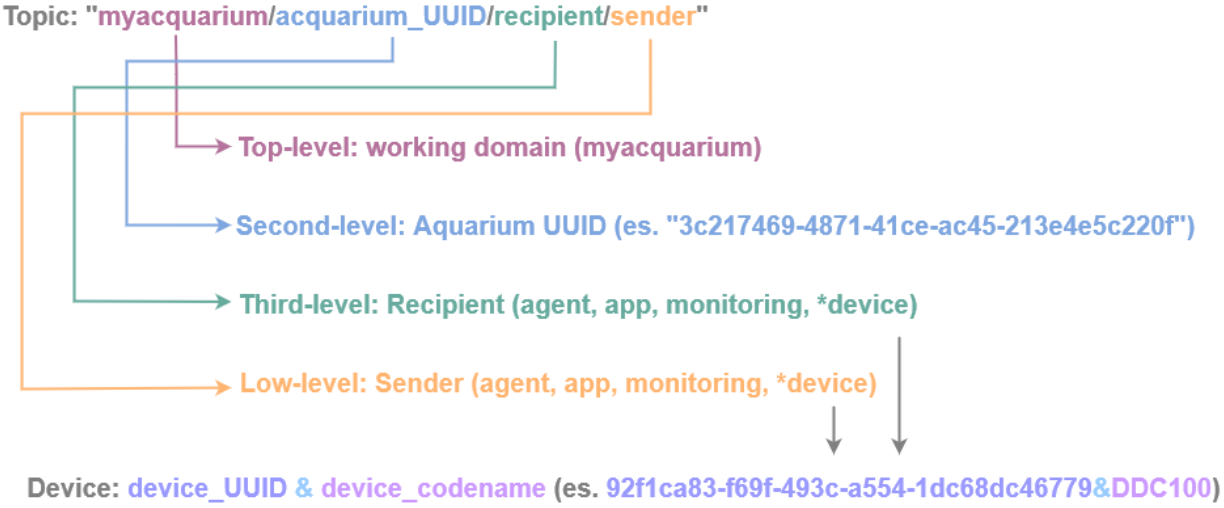
L’autenticazione con Keycloak avviene utilizzando mTLS. Questa opzione è disattivata per impostazione predefinita, ma attivata configurando la variabile “KC_HTTPS_CLIENT_AUTH” a “required”. Il file “keycloak-keystore.jks” contiene le informazioni delle chiavi e del certificato di Keycloak, ed è protetto dalla password “password”. Il file “my-truststore.jks”, anch’esso protetto dalla password “password”, contiene le informazioni dei certificati di cui fidarsi: verranno ammessi soltanto i client con certificati fidati. Il reame utilizzato dal servizio viene caricato automaticamente col comando “--import-realm” se viene inserito il file di backup “realm.json” nella cartella “/opt/keycloak/data/import”. Così facendo il container è pronto all’utilizzo, senza dover ricreare o importare il backup successivamente.

Il terzo comando genera un container di nome “mosquitto” sull’immagine “eclipse-mosquitto” di versione “2.0.15”. Si apre la porta 8883 per le comunicazioni, che avvengono tramite protocollo TLSv1.2.

Viene passato il file “mosquitto.conf” per modificare la configurazione di default di Mosquitto. Il file configura gli aspetti presenti nell’immagine seguente.



Il file “acl_file.txt” contiene regole per controllare l’acceso dei client ai topics del broker.

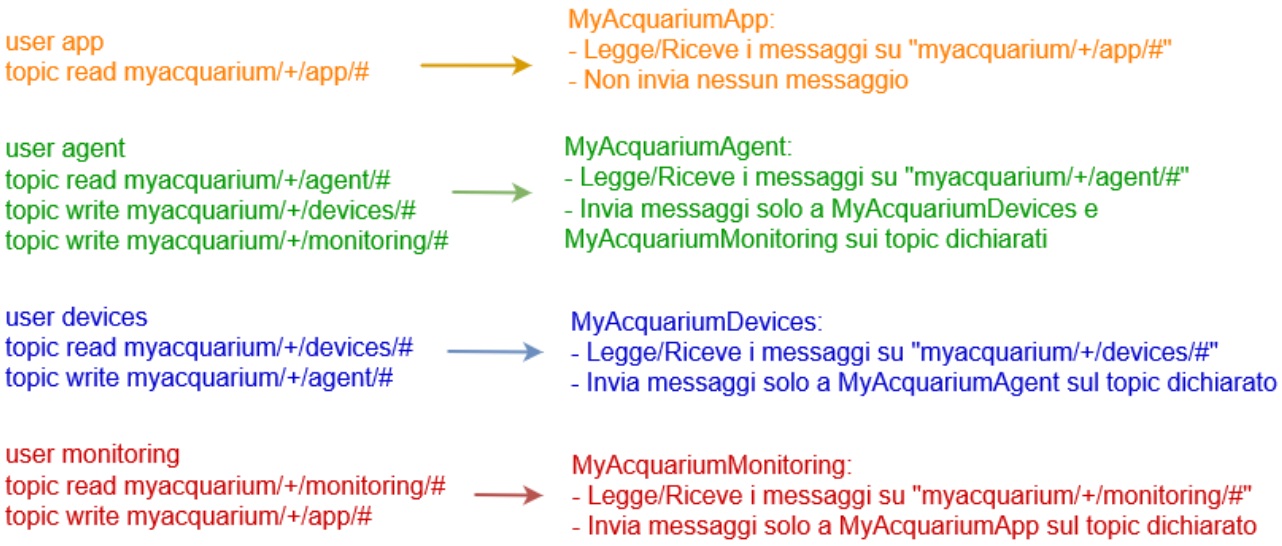



I topics, come è possibile notare dall’immagine, sono composti da:

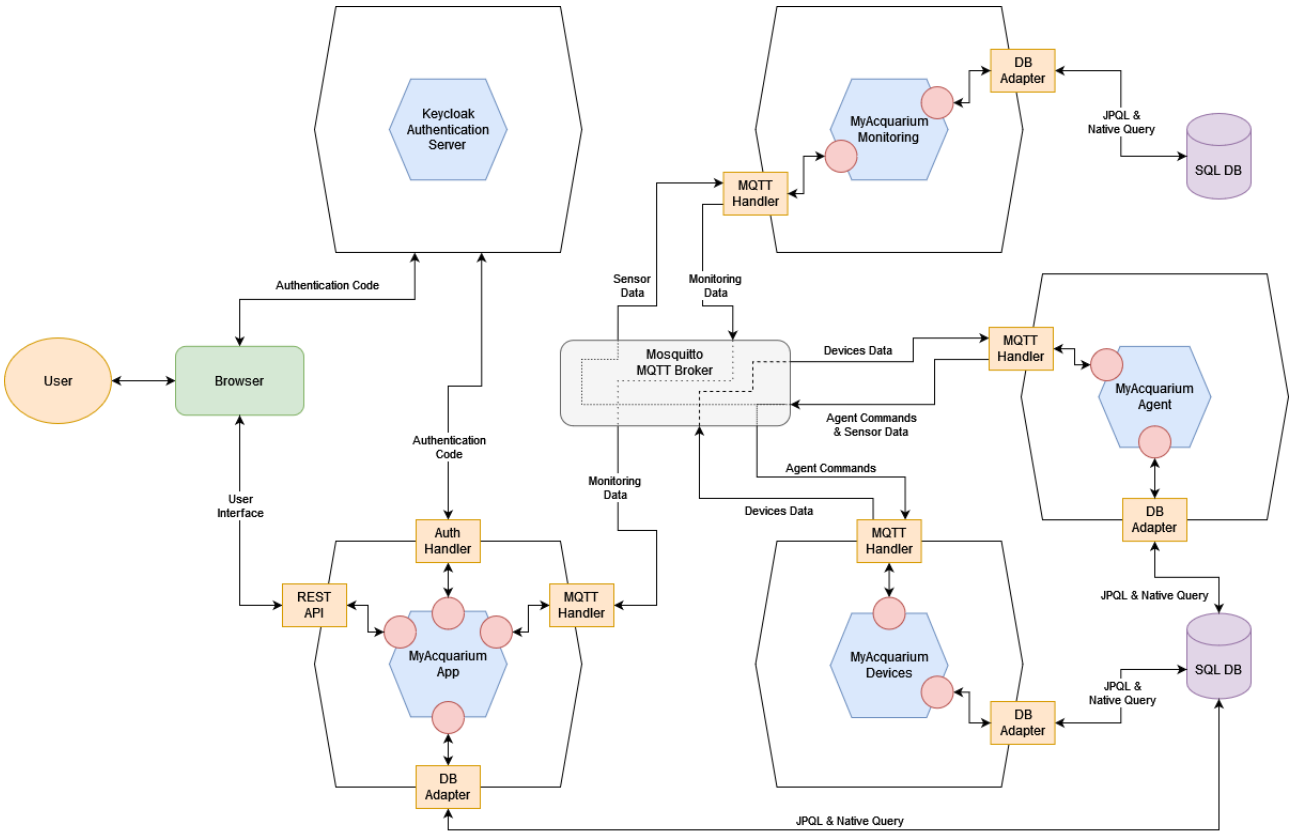
- “myacquarium”, top-level sempre presente per indicare il nome del servizio MyAcquarium;
- UUID dell’acquario, per indicare l’acquario di riferimento;
- destinatario, che può prendere la forma di:
  - “agent”, se il messaggio è diretto a MyAcquariumAgent;
  - “app”, se il messaggio è diretto a MyAcquariumApp;
  - “monitoring”, se il messaggio è diretto a MyAcquariumMonitoring;
  - UUID e codice del device, se il messaggio è diretto a MyAcquariumDevices;
- mittente, che può prendere le stesse forme del destinatario (vedi sopra).

I messaggi inviati dai dispositivi, o per i dispositivi, utilizzano la forma “UUID & codice del dispositivo”.

Ritornando al file “acl_file.txt”, le regole di accesso sono le seguenti.



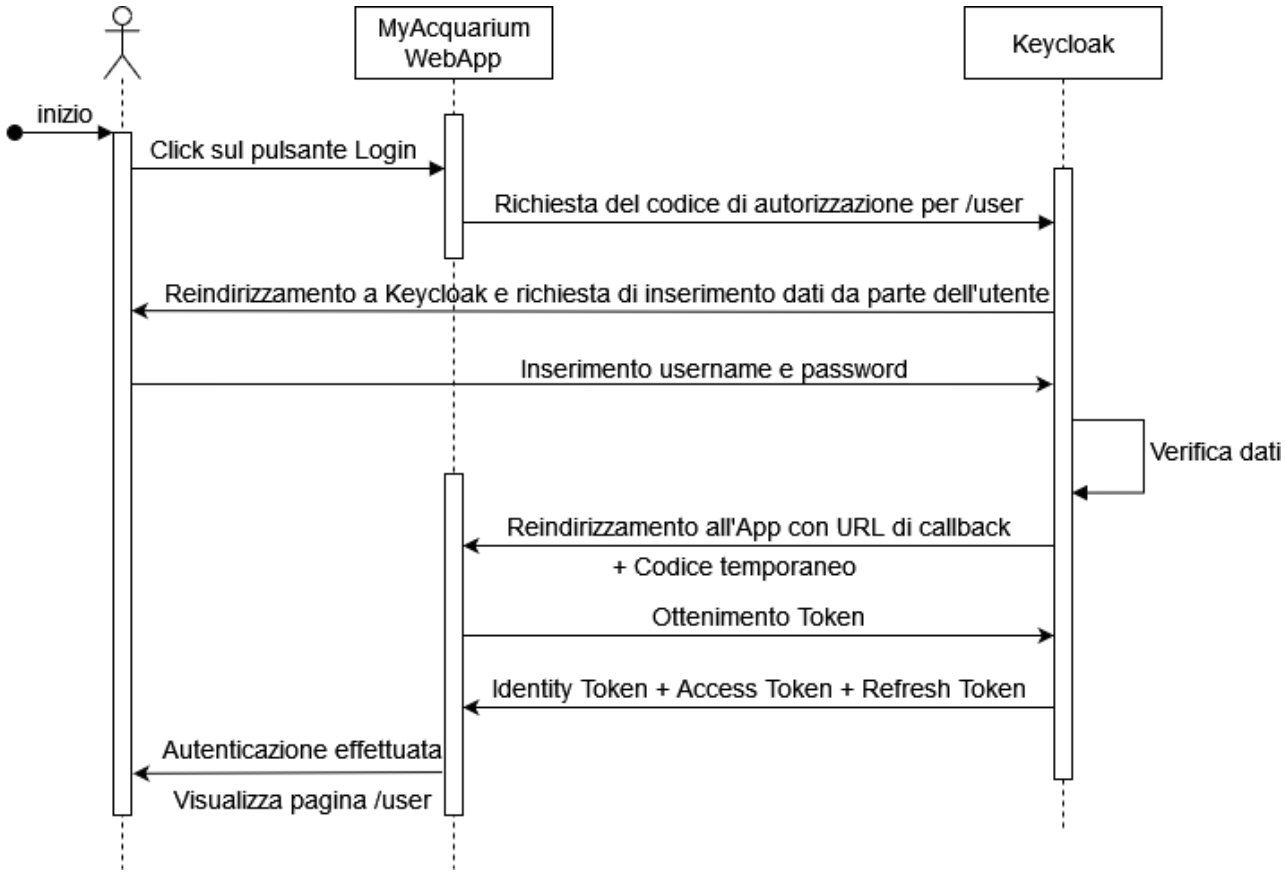
## Comunicazione tra microservizi




I microservizi, come già specificato precedentemente, comunicano tra di loro mediante l'invio e la ricezione di messaggi MQTT attraverso un message broker. La comunicazione avviene su protocollo TLS, e i client devono presentare un certificato valido per la connessione. Ogni microservizio ha i propri certificati salvati nella cartella "certificates". La parte del nome del file indicata con "nomeservizio" varia a seconda del microservizio (app per MyAcquariumApp, agent per MyAcquariumAgent, devices per MyAcquariumDevices, e monitoring per MyAcquariumMonitoring):

- "MQTT_nomeservizio.crt", il certificato personale in formato crt (Certificate File Format);
- "MQTT_nomeservizio.key", la chiave privata in formato key (contiene la chiave privata salvata nello standard RSA PKCS#1);
- "MQTT_ca.crt", il certificato della Certification Authority che ha emesso il certificato personale, ed è la stessa che ha emesso il certificato del message broker.

La comunicazione con Keycloak avviene per autenticare l'utente attraverso lo standard OpenID Connect. Viene abilitato il flusso standard per l'autenticazione, chiamato Authorization Code Flow:




- L'utente si collega all'applicazione utilizzando un browser e prova ad accedere a risorse protette. Nel nostro caso, supponiamo prema il tasto di "login", che porta alla pagina dei dati dell'utente, accessibile solo da un utente autenticato correttamente.
- L'applicazione reindirizza il browser a Keycloak per l'autenticazione. L'applicazione passa un URL di callback come parametro di query nel reindirizzamento del browser. Keycloak utilizza il parametro al termine dell'autenticazione.
- L'utente inserisce il proprio username e la propria password. Se corretti, Keycloak autentica l'utente e crea un codice temporaneo di breve durata.
- Keycloak reindirizza all'applicazione utilizzando l'URL di callback e aggiunge il codice temporaneo come parametro di query nell'URL di callback.
- L'applicazione estrae il codice temporaneo ed effettua un'invocazione REST in background a Keycloak per scambiare il codice con token d'identità, di accesso, e di aggiornamento. Per evitare attacchi replay, il codice temporaneo non può essere utilizzato più di una volta.

Viene inoltre abilitato il supporto al Direct Access Grants per garantire all'applicazione l'accesso all'username e alla password dell'utente così da scambiarle direttamente a Keycloak per ottenere i token. Si tratta quindi di una richiesta HTTP POST contenente le credenziali dell'utente e l'id del client per ottenere l'Identity Token, l'Access Token, e il Refresh Token.

L'Identity Token contiene informazioni sull'utente, come il suo username, il nome, il cognome, l'email, e altre informazioni sul suo profilo.

L'Access Token contiene informazioni sull'accesso che l'applicazione può utilizzare per determinare quali risorse l'utente è autorizzato ad accedere. Nel nostro caso, l'utente è registrato col ruolo "user" e può accedere alle risorse del sistema MyAcquarium protette per soli utenti con ruolo "user".

Il Refresh Token viene utilizzato per ottenere un nuovo Access Token quando esso è scaduto, in quanto ha una durata di soli 5 minuti, mentre il Refresh Token ha una durata di 30 minuti. Alla scadenza del Refresh Token (dopo quindi 30 minuti di inattività) l'utente verrà reindirizzato a Keycloak per una nuova autenticazione.

## Approccio tecnologico

La struttura è suddivisa in microservizi per migliorare le prestazioni poiché è possibile gestire singolarmente i microservizi piuttosto che l'intera applicazione nel suo complesso. C'è inoltre una maggiore sicurezza dei dati, una maggiore scalabilità, e un'integrazione continua.

Spring Boot è stato scelto per creare applicazioni con facilità e con molta meno fatica rispetto ad altri paradigmi grazie alle sue numerose integrazioni e configurazioni automatiche, per esempio Spring Data JPA, che grazie all'utilizzo delle Java JPA e di Hibernate, consente una semplificazione del dialetto SQL pensando sottoforma di entità Java piuttosto che di tabelle e colonne SQL native.

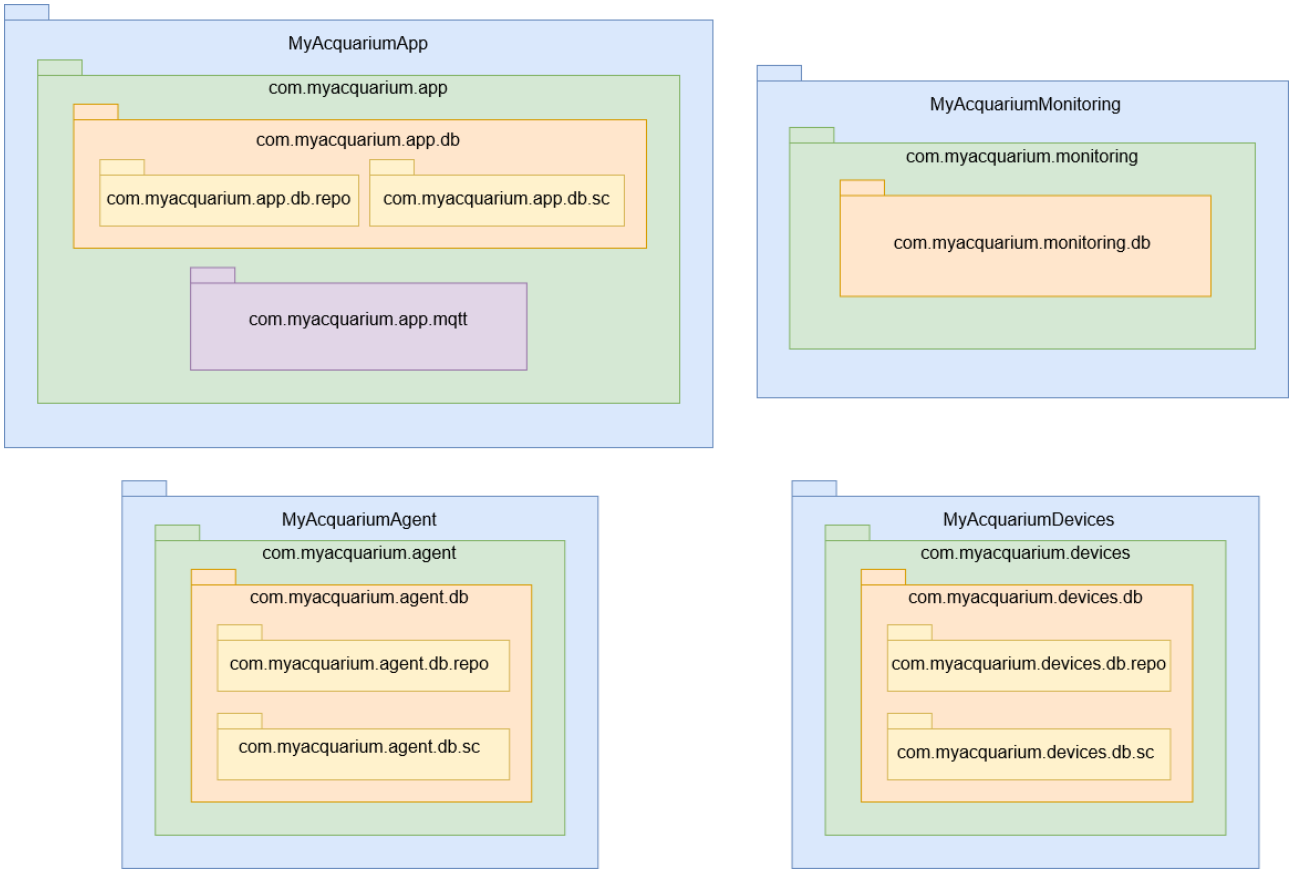
MySQL è il sistema di gestione di database relazionali preferito per la sua facilità d'uso, velocità, compatibilità, scalabilità, e sicurezza.

Keycloak è uno strumento veloce per gestire l'autenticazione e l'autorizzazione dei propri utenti a servizi offerti dall'applicazione.

MQTT è più facile da configurare rispetto ad altri protocolli di messaggistica, e il sistema publish-subscribe consente a molti client di produrre e condividere informazioni tra loro, perfetto nel nostro campo per comunicare con i device o tra i vari microservizi. Mosquitto è il message broker utilizzato che implementa il protocollo MQTT.

Docker ha permesso di mantenere i database, il server Keycloak, e il message broker Mosquitto, in container separati gestibili, attivabili, e movibili singolarmente per una facilità di sviluppo ancora maggiore.

## Struttura moduli e package




I microservizi sono suddivisi in package per organizzare le classi Java a seconda del loro contesto.

Il package principale, colorato in verde, possiede classi necessarie per il lancio dell'applicazione Spring. In MyAcquariumApp serve inoltre a gestire le richieste API REST, mentre negli altri microservizi fornisce classi per il supporto del protocollo MQTT. Anche l'applicazione ha bisogno del supporto MQTT, che è quindi presente nel subpackage "mqtt" del package principale.

I subpackage "db" del package principale, colorati in arancione, forniscono le classi persistenti delle entità che rappresentano gli oggetti del database. I subpackage "repo" del package "db" forniscono le interfacce per le operazioni CRUD (Create, Read, Update, Delete) generiche sui repository, mentre i subpackage "sc" forniscono classi astratte designate come MappedSuperclass le cui informazioni di mappatura sono applicate alle entità che ereditano da esse, per evitare di scrivere codice ripetuto in ogni classe delle entità. MyAcquariumMonitoring non possiede quest'ulteriore divisione in "repo" e "sc" in quanto le classi sono di numero inferiore ed è tutto all'interno del package "com.myacquarium.monitoring.db".

## Documentazione JavaDoc

Per ogni microservizio del sistema MyAcquarium è disponibile la documentazione JavaDoc di visibilità privata (contenente tutte le classi e i membri) che si preoccupa di dare informazioni sul codice Java.

# Implementazione

## Maven Project Object Model

Come già citato in precedenza, anche i microservizi, essendo progetti Maven, possiedono un file XML denominato “pom.xml” che definisce il nome del progetto, le versioni e le dipendenze sulle librerie esterne. In tutti i microservizi sono presenti le dipendenze necessarie per Spring Boot, JUnit Test, MQTT, Spring Data JPA, Hibernate, e MySQL. In MyAcquariumApp sono presenti dipendenze aggiuntive necessarie alla WebApp, ovvero Thymeleaf, Spring Security, e Keycloak.

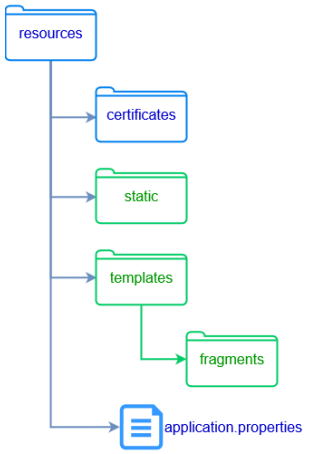
## Resources

Nella cartella “resources” sono salvate risorse utili al microservizio. Tutti presentano la sottocartella “certificates” contenente i certificati, e il file “application.properties” per configurare proprietà del servizio. MyAcquariumApp possiede inoltre file statici (es. immagini) nella sottocartella “static”, template di Thymeleaf nella cartella “templates” con al suo interno dei frammenti di pagina nella cartella “fragments”.

Il file “application.properties” imposta per ogni microservizio il collegamento col DB. È necessario conoscere il Java DataBase Connectivity (JDBC) URL del database, l’username, e la password di accesso. Nell’URL JDBC viene indicato di creare il DB se non esiste. Inoltre si salva il tipo di driver JDBC da utilizzare, MySQL, anche se è possibile ricavarlo automaticamente dall’URL dato in precedenza. In più, si indica di inizializzare lo schema all’avvio, o di aggiornarlo in mancanza di alcune tabelle, e viene fatto notare ad Hibernate che si utilizza un DB MySQL di versione 8 o superiore, per comunicare con query SQL appropriate.

In MyAcquariumApp, ulteriormente, viene impostato il numero di porta, la 8444, per il server HTTP. Viene segnalato di attivare il supporto SSL avvalendosi del protocollo TLSv1.2 e vengono indicati i certificati da utilizzare per mTLS come già spiegato precedentemente. Sono presenti proprietà per il collegamento col client “myacquarium-app” del reame “MyAcquarium” di Keycloak, indicando sempre i file dei certificati per la mutua autenticazione.

A seguire, un’immagine per riassumere la struttura delle cartelle e il file “application.properties”. In blu sono indicate le cartelle e i valori presenti in ogni microservizio, in verde solo quelli di MyAcquariumApp. Da notare il cambio di DB per il microservizio MyAcquariumMonitoring indicato in rosso.




```
server.port=8444

server.ssl.enabled=true
server.ssl.key-store-type=JKS
server.ssl.key-store=classpath:certificates/app-keystore.jks
server.ssl.key-store-password=password
server.ssl.protocol=TLS
server.ssl.enabled-protocols=TLSv1.2

server.ssl.client-auth=need
server.ssl.trust-store=classpath:certificates/app-truststore.jks
server.ssl.trust-store-password=password

keycloak.realm=MyAcquarium
keycloak.auth-server-url=https://localhost:8443/
keycloak.ssl-required=all
keycloak.resource=myacquarium-app
keycloak.public-client=true
keycloak.confidential-port=8443
keycloak.principal-attribute=preferred_username

keycloak.truststore=classpath:certificates/app-truststore.jks
keycloak.truststore-password=password

keycloak.client-keystore=classpath:certificates/app-keystore.jks
keycloak.client-keystore-password=password
keycloak.client-key-password=password

spring.mvc.hiddenmethod.filter.enabled=true

spring.datasource.url=jdbc:mysql://localhost:3306/myacquarium?createDatabaseIfNotExist=true
spring.datasource.username=root                ...3306/monitoring?create...
spring.datasource.password=root
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.jpa.generate-ddl=true
spring.jpa.show-sql=false
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
spring.jpa.hibernate.ddl-auto=update
spring.jpa.open-in-view=true
```

## Java Classes

Nella cartella “java” sono presenti le classi Java suddivise in vari package. Nel package principale, la classe che prende il nome dal microservizio (es. MyAcquariumApp), serve per eseguire il bootstrap e lanciare l’applicazione.

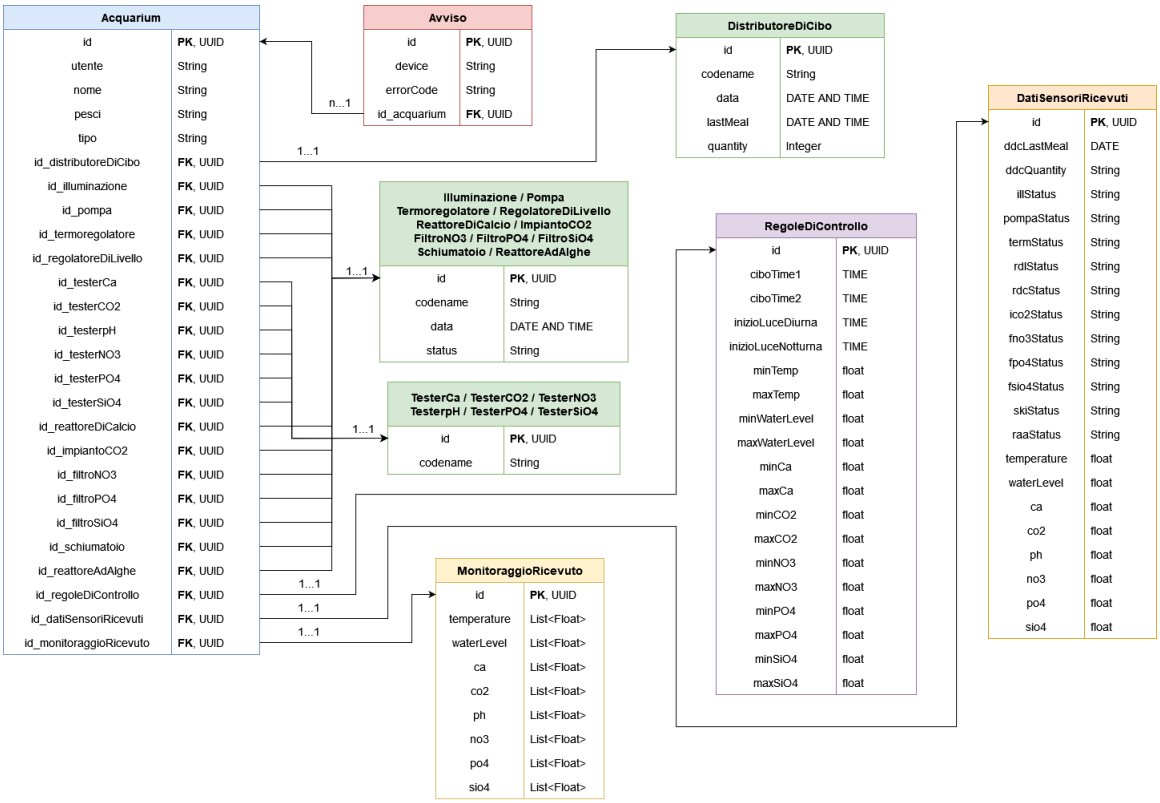
Sempre all’interno del package principale, MyAcquariumApp presenta la classe SecurityConfig per la configurazione di Spring Security e KeycloakConfig per la configurazione di Keycloak. MyController è la classe responsabile dell'elaborazione delle richieste API REST in arrivo, della preparazione di un modello, e della restituzione della vista da rendere come risposta. Le pagine sono scritte in linguaggio HTML con aggiunta di codice riferito al template engine Thymeleaf che verrà da esso elaborato per produrre le pagine visualizzate dall’utente. Viene utilizzata una classe di supporto che prende il nome di AcquariumForm per salvare gli input dell’utente nei form di creazione o modifica dell’acquario e convertirti nell’effettiva classe dell’entità Acquario.

## Entities e Repositories

JPA permette di definire classi Java come entità che raffigurano tabelle persistenti in un database dove ogni istanza della classe rappresenta una riga nella tabella. MyAcquariumApp, MyAcquariumAgent, e MyAcquariumDevices, poiché utilizzano lo stesso database, possiedono le stesse classi delle entità e le stesse classi repository e MappedSuperclass presenti rispettivamente nei subpackage “db”, “repo”, e “sc”.

Sono presenti 2 MappedSuperclass: EntityWithId, che serve ad applicare l’informazione di UUID (Universal Unique Identifier) ad ogni altra entità, e Device, che serve ad applicare l’informazione del nome e della data alle sole entità che rappresentano i dispositivi.

Il diagramma seguente mostra uno schema delle classi entità di MyAcquariumApp, MyAcquariumAgent, e MyAcquariumDevices. Per semplicità, le informazioni delle MappedSuperclass sono già state applicate alle entità, e le tabelle di entità aventi gli stessi attributi sono state inglobale in un’unica tabella.



MyAcquariumMonitoring utilizza un database separato per salvare i valori ricevuti tenendo traccia delle serie storiche. È presente una sola classe "Monitoraggio": il nuovo valore ricevuto viene aggiunto alla fine della lista. Ogni lista può contenere al massimo 10 valori, dopodiché il valore più vecchio (situato in cima alla lista) viene eliminato per fare spazio al valore più recente.

| Monitoraggio | |
|---|---|
| id | **PK**, UUID |
| acquariumId | UUID |
| temperature | List<Float> |
| waterLevel | List<Float> |
| ca | List<Float> |
| co2 | List<Float> |
| ph | List<Float> |
| no3 | List<Float> |
| po4 | List<Float> |
| sio4 | List<Float> |

Le classi repository, implementate da Spring, sono richiamate nelle varie classi del servizio per poter accedere alle operazioni CRUD della gestione persistente dei dati. È possibile estendere ulteriormente le query disponibili creando metodi con keywords particolari supportate da JPA: un esempio è il metodo "List<Acquarium> findByUtente(String username)" presente nella repository Acquarium che viene tradotto automaticamente nella query "select a from Acquarium a where a.utente = ?1".

## REST Controller e Thymeleaf

La classe MyAcquariumApplication presenta metodi necessari al lancio dell'applicazione Spring e alla configurazione di Spring Security e Keycloak. In particolare, vengono protetti gli indirizzi interni dell'applicazione (percorsi /user, /acquariums, /warnings, /fishes, e loro sottopercorsi) ai soli utenti di ruolo "user". La pagina principale è disponibile, naturalmente, a tutti i visitatori.

Le pagine sono scritte in linguaggio HTML con aggiunta di codice riferito al template engine Thymeleaf che verrà da esso elaborato per produrre le pagine visualizzate dall'utente.

Nella classe MyController sono presenti metodi che gestiscono richieste HTTP. Le richieste HTTP GET sono:

- Visualizzazione Home Page: richiesta al path "/"
  - Ritorna semplicemente la pagina principale chiamata "index.html".
- Logout: richiesta al path "/logout"
  - L'utente viene disconnesso e riportato alla pagina principale (index.html).
- Visualizzazione pagina dati utente: richiesta al path "/user"
  - Ottiene l'Access Token da Keycloak per prelevare i dati dell'utente (nome, cognome, email) da aggiungere come attributi al modello della pagina assieme a un valore booleano per indicare se l'utente ha già creato degli acquari. Viene restituita la pagina dei dati dell'utente denominata "user.html" una volta aggiunti tutti gli attributi necessari a Thymeleaf per generarla.
- Visualizzazione pagina acquari: richiesta al path "/acquariums"
  - Come per la richiesta GET a "/user", anche qui si prelevano i dati dell'utente dall'Access Token da aggiungere come attributi al modello assieme alla lista di acquari dell'utente e al

valore booleano per indicare se ha già creato degli acquari. La pagina “acquariums.html” viene ritornata una volta aggiunti gli attributi del modello necessari.

- Visualizzazione pagina dettagli acquario: richiesta al path “/acquariums/{id}”
  - “{id}” è una variabile dell’URI che indica l’UUID dell’acquario mostrato nella pagina. Di conseguenza, si ritorna la pagina “acquariumInfo.html” dopo aver aggiunto al modello l’acquario di UUID dichiarato nell’URI. Se non è disponibile nessun acquario viene mostrata la pagina di errore “error.html”.
- Visualizzazione pagina creazione acquario: richiesta al path “/acquariums/new”
  - Al modello viene aggiunta una nuova istanza della classe AcquariumForm necessaria per salvare gli input dell’utente riguardo all’aggiunta di un nuovo acquario, e l’attributo booleano “editMode” impostato a falso per indicare a Thymeleaf di generare la pagina per la creazione di un nuovo acquario, in quanto viene utilizzato lo stesso modello per generare anche la pagina di modifica dell’acquario. Terminato, viene restituita la pagina “newOrEditAcquarium.html”.
- Visualizzazione pagina modifica dispositivi: richiesta al path “/acquariums/{id}/editDevices”
  - La variabile id dell’URI viene utilizzata per cercare l’acquario e salvare il suo nome, il suo tipo, e le tipologie di pesci, come attributi al modello. Viene inoltre aggiunta un’istanza della classe AcquariumForm per salvare gli input di modifica dell’acquario, e l’attributo booleano “editMode” impostato a true per indicare a Thymeleaf di generare la pagina di modifica di un nuovo acquario. Viene restituita la pagina “newOrEditAcquarium.html”.
- Visualizza pagina modifica regole: richiesta al path “/acquariums/{id}/editRules”
  - La variabile id dell’URI viene utilizzata per cercare l’acquario e salvare il suo nome, il suo tipo, e le regole di gestione, come attributi al modello prima di restituire la pagina “editRules.html”.
- Visualizzazione pagina tipologie di pesci: richiesta al path “/fishes”
  - Ritorna semplicemente la pagina “fishes.html”.
- Visualizzazione pagina avvisi: richiesta al path “/warnings”
  - Al modello della pagina “warnings.html” ritornata vengono aggiunti gli acquari dell’utente e la variabile booleana che indica se l’utente possiede già degli acquari.

È presente un metodo che risponde all’unica richiesta HTTP POST su “/acquariums/new” per la creazione di un nuovo acquario e reindirizzare successivamente l’utente alla pagina degli acquari posseduti. Il metodo crea un nuovo acquario e lo imposta secondo i valori dichiarati dall’utente salvati sul form. Gli vengono aggiunte le regole di controllo predefinite secondo il tipo di acquario e riferimenti per salvare successivamente i dati ricevuti dai sensori e il monitoraggio.

Due metodi gestiscono richieste HTTP PUT su "/acquariums/{id}/editDevices" e "/acquariums/{id}/editRules" per fornire rispettivamente il supporto alla modifica dell’acquario e alle regole di gestione. L’id nell’URI indica l’acquario in fase di modifica. Per la modifica dell’acquario viene utilizzato il form di supporto per salvare gli input dell’utente e riflettere le modifiche sull’acquario. Gli input sulle regole di gestione alterano già le regole effettive. Terminata la modifica, l’utente viene riportato, in entrambi i casi, alla pagina degli acquari. Se la modifica non va a buon fine o non è disponibile nessun acquario di id specificato viene mostrata la pagina di errore “error.html”.

Per concludere, altri due metodi assistono richieste HTTP DELETE su “/acquariums/{id}” e “/warnings/{id}” per l’eliminazione di un acquario o di un avviso. L’id nell’URI indica l’acquario o l’avviso da eliminare. Dopo aver eliminato correttamente l’acquario si ritorna alla pagina degli acquari, mentre dopo l’eliminazione di un avviso si ritorna alla pagina degli avvisi. La pagina di errore viene mostrata in caso di eliminazione non avvenuta correttamente o id non valido.

Di seguito uno schema che riassume i controller elencati precedentemente.

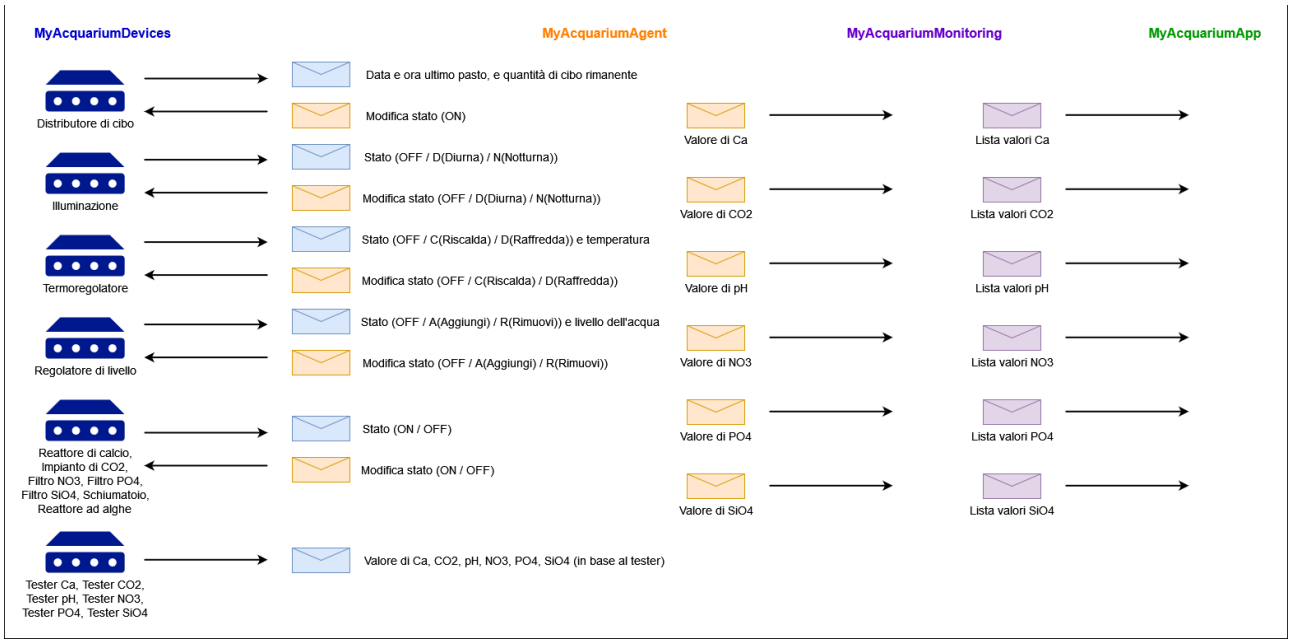
| HTTP GET | |
|---|---|
| / | Home Page |
| /logout | Richiesta Logout |
| /fishes | Pagina tipologie di pesci |
| /user | Pagina dati utente |
| /warnings | Pagina avvisi |
| /acquariums | Pagina acquari |
| /acquariums/new | Pagina creazione acquario |
| /acquariums/{id} | Pagina dettagli acquario |
| /acquariums/{id}/editDevices | Pagina modifica dispositivi |
| /acquariums/{id}/editRules | Pagina modifica regole |

| HTTP POST | |
|---|---|
| /acquariums/new | Creazione acquario |

| HTTP PUT | |
|---|---|
| /acquariums/{id}/editDevices | Modifica dispositivi |
| /acquariums/{id}/editRules | Modifica regole |

| HTTP DELETE | |
|---|---|
| /acquariums/{id} | Elimina acquario |
| /warnings/{id} | Elimina avviso |

## MQTT Messages




La classe MqttBeans, presente in ogni microservizio, configura gli adattatori di canali in entrata e in uscita per il supporto al protocollo MQTT. Viene configurato un client MQTT per la corretta connessione al message broker, passando i certificati per l'apertura di un socket SSL, definendo i criteri per l'invio dei messaggi, e costruendo le interfacce per i messaggi in entrata e in uscita. Il metodo "public MessageHandler handler()" gestisce i messaggi MQTT in arrivo.

I messaggi vengono inviati con l'ausilio delle classi MyGateway, che fornisce un gateway di messaggistica definendo un metodo per inviare un messaggio MQTT, e MqttSender, dove sono presenti task schedulati per l'inoltro di messaggi dopo un certo periodo di tempo prefissato. MyAcquariumApp non possiede queste due classi poiché non invia messaggi ma riceve le serie storiche da MyAcquariumMonitoring.

MyAcquariumDevices invia messaggi all'agente ogni minuto. Tutti i dispositivi dell'acquario, ogni minuto, inviano il proprio stato e i valori dell'acquario. In caso di malfunzionamento inviano messaggi che

comprendono il codice di errore da segnalare all'utente che provvederà a risolvere manualmente. Ricevono messaggi unicamente dall'agente con comandi per alterare il loro stato (es. attivare il distributore di cibo).

MyAcquariumAgent riceve i valori inviati dai dispositivi, li salva, e in base alle regole di gestione calcola i dispositivi da attivare o disattivare. I messaggi vengono inviati alla ricezione dei valori dei dispositivi: non vengono inoltrati messaggi dopo un periodo di tempo prefissato, pertanto la classe MqttSender non è presente. In aggiunta, si trasmettono i valori dei dispositivi a MyAcquariumMonitoring per il monitoraggio.

MyAcquariumMonitoring, come riferito precedentemente, riceve i valori dei dispositivi dall'agente, e possiede la classe MqttSender per inviare ogni 5 minuti a MyAcquariumApp la lista delle serie storiche dei valori dell'acquario.

## Validazione

Per verificare il corretto funzionamento del sistema sono stati verificati:

- Collegamenti con MySQL, Mosquitto, e Keycloak
- Controller REST
    - Test di correttezza delle pagine
    - Test sulle chiamate GET / POST / PUT / DELETE
    - Test di eventuali errori
- Database
    - Controllo delle tabelle
    - Controllo degli attributi
- Inoltro e ricezione di messaggi
    - Analisi dei messaggi inoltrati e ricevuti
    - Controllo sull'inoltro randomico dei valori dei dispositivi
    - Test sul comportamento dovuto all'inoltro di messaggi generati ad hoc
- Agente
    - Verifica della correttezza dell'invio di messaggi secondo le regole di gestione
- File jar eseguibili

## Deployment

Il servizio MyAcquarium, grazie alla divisione in microservizi Spring Boot, comprende file jar eseguibili pronti per essere dislocati sui più diffusi fornitori di cloud PaaS (Platform-as-a-Service) come Cloud Foundry, Heroku, Amazon Web Services (AWS), Azure, o Google Cloud.

Sono presenti inoltre file jar completamente eseguibili per sistemi Unix.