Progetto Programmazione Avanzata

Sandro Wu

A.A 22-23

Indice

1 Panoramica

1.1 Idea del progetto

L'idea dell'applicazione è un Manager di servizi, in questo caso Servizi Cloud (ispirazione da Microsoft Azure). In breve l'applicazione offre la possibilità per l'utente di visualizzare, creare, eliminare i propri servizi e modificarne lo stato, visualizzare i log delle azioni eseguite ed i costi sostenuti per i servizi.

L'applicazione necessita di dati forniti dall'esterno. Per esempio i dati dei servizi offerti dalla piattaforma cloud, i dati dei costi fatturati all'utente.

1.2 Inizializzazione Database

Il progetto è composto da due repository Applicazione e Server.
Creare prima il database dall'Applicazione. Solo dopo ciò sarà possibile avviare il Server.

Quindi, per inizializzare il database:

- All'avvio dell'Applicazione, comparirà la schermata di login.
- E' presente un bottone in basso a destra "Initialize database".
- Cliccandoci si crea (e sovrascrive) il database e si popola con i dati predefiniti.

L'Applicazione si connette direttamente a MySQL con parametri:

```
IP: 127.0.0.1
Porta: 3306
Database: <Matricola>
Username : root
Password : root
```

I file coinvolti sono:

```
> src/main/java/progetto/applicazione/InitDB.java
> src/main/resources/progetto/applicazione/data.json
> src/main/resources/progetto/applicazione/create_table.sql
```

Il database creato ha per semplicità tabelle legati con dei "falsi" foreign keys, quindi di fatto indipendenti tra loro. Lo scopo è usare così Spring JPA solo a livello di base, senza entrare a livelli troppo tecnici, poiché fuori dall'obiettivo del progetto.

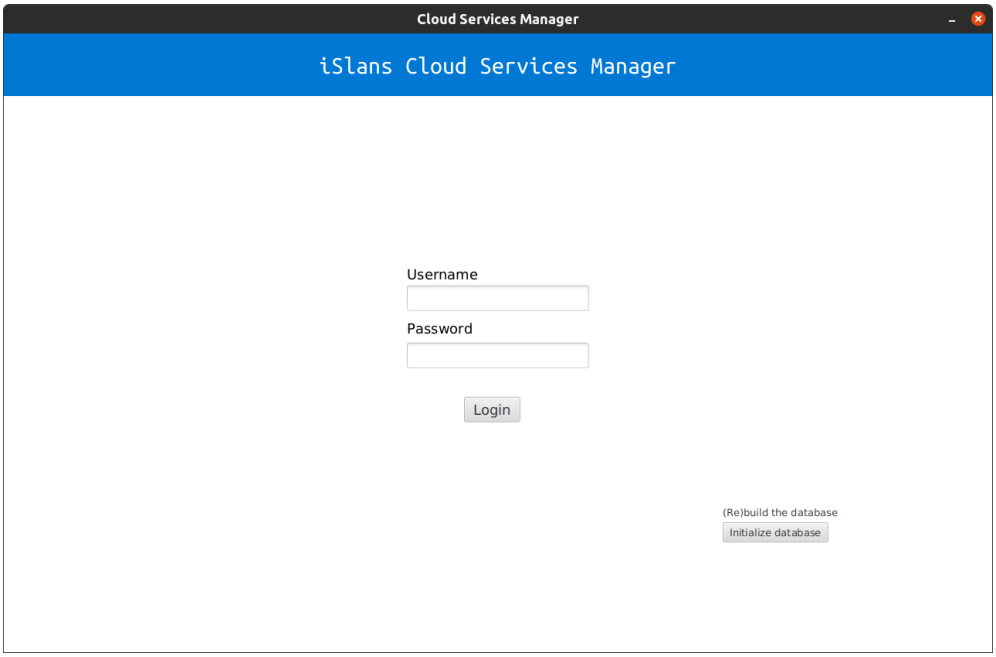
2 Applicazione

2.1 Schermate dell'applicazione

2.1.1 Login Page

All'avvio dell'applicazione è richiesto il login. Non è previsto la registrazione di nuovi utenti dall'applicazione. L'account di test è

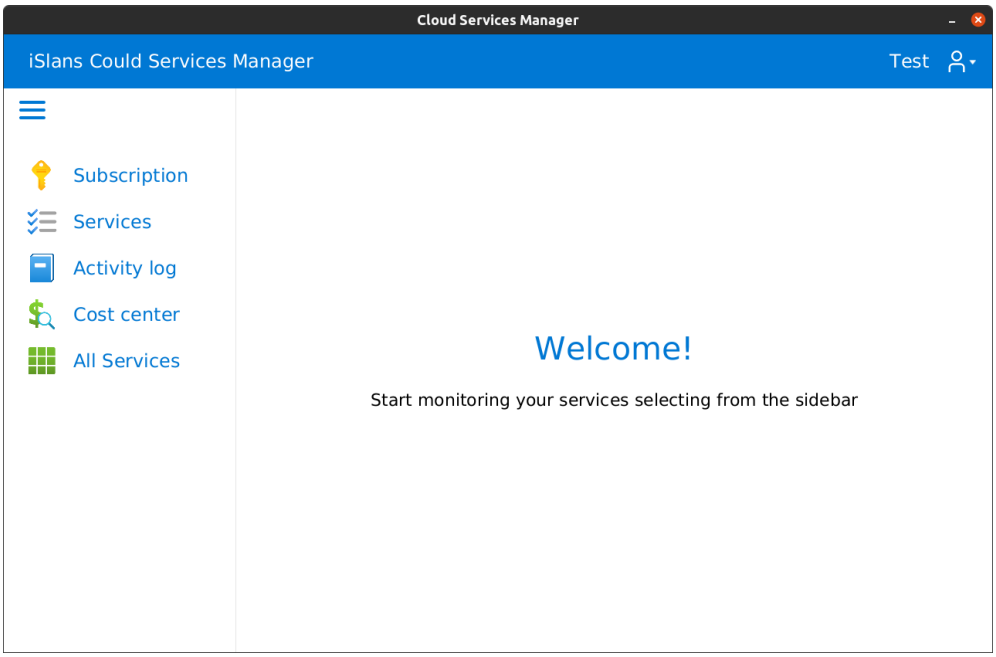
```
User: Test
Pass: test
```



2.1.2 Home Page

L'interfaccia è composta da:

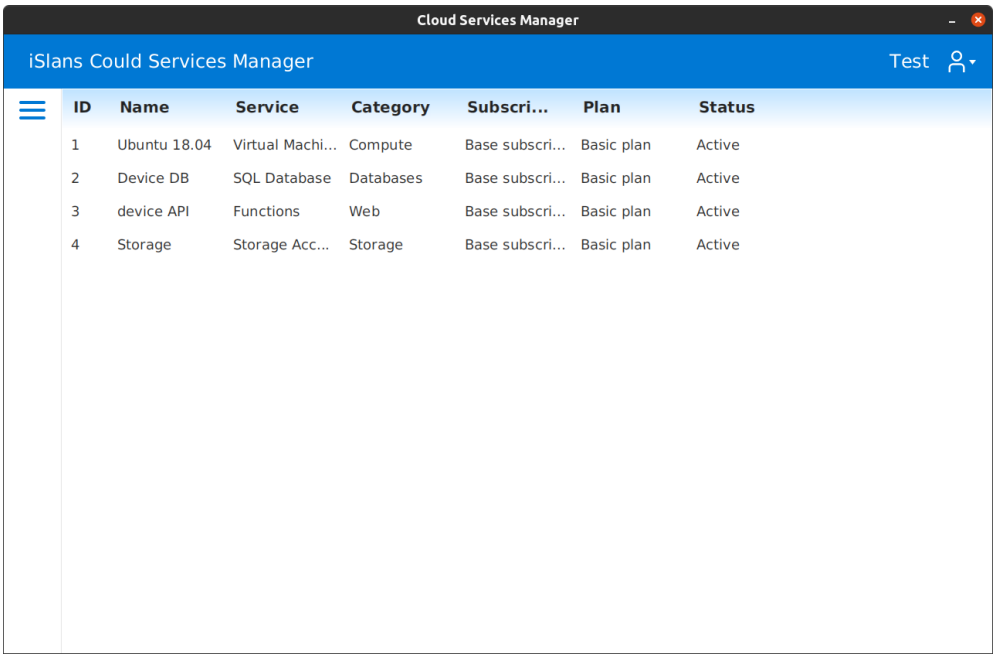
- TopBar, la barra in alto. Mostra a destra l'username e un'icona. L'icona contiene un menù a scomparsa con l'opzione di logout.
- SideBar, a sinistra. Contiene un menù ad hambuger per espandersi/chiudersi, ed i vari link per le pagine dell'applicazione
- MainBody è tutta la parte restante, che viene caricato dinamicamente in base alla pagina selezionata nel SideBar

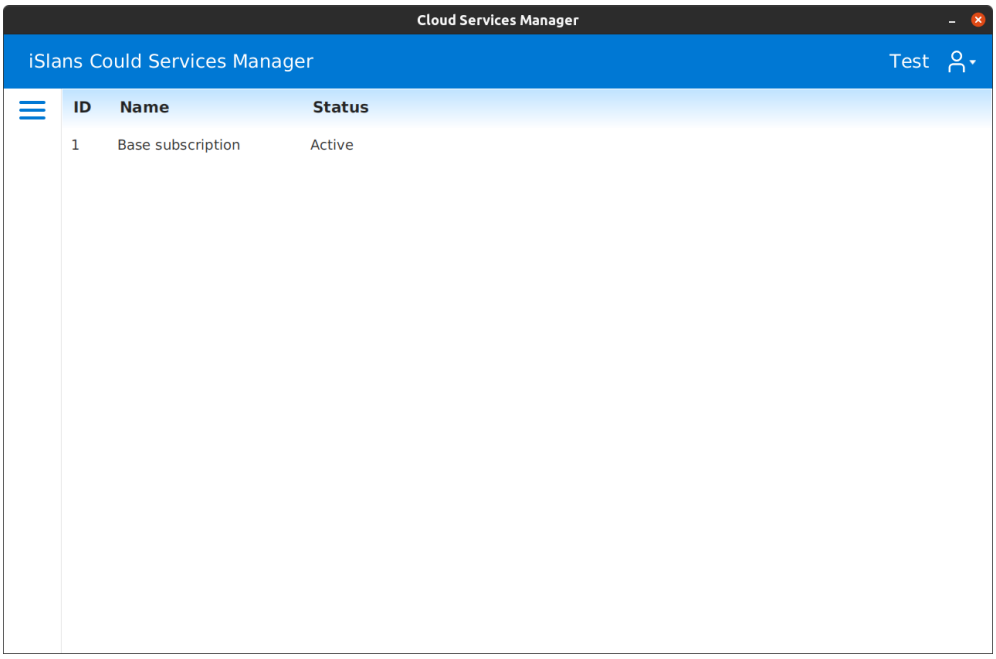


2.1.3 Services e Subscription

Qui vi sono i servizi e sottoscrizioni dell'utente. Attraverso il menù a scomparsa, si può

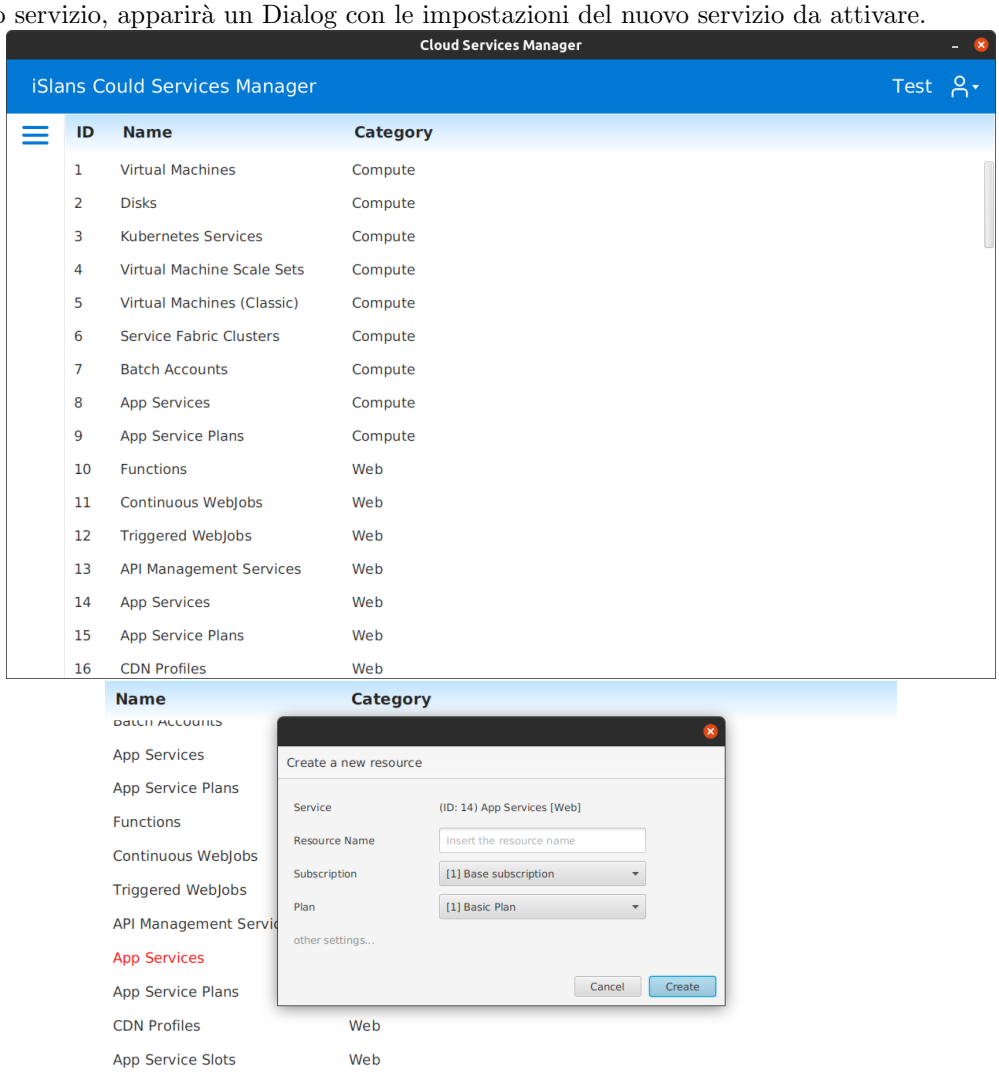
- attivare / fermare / eliminare i servizi
- attivare / fermare le sottoscrizioni





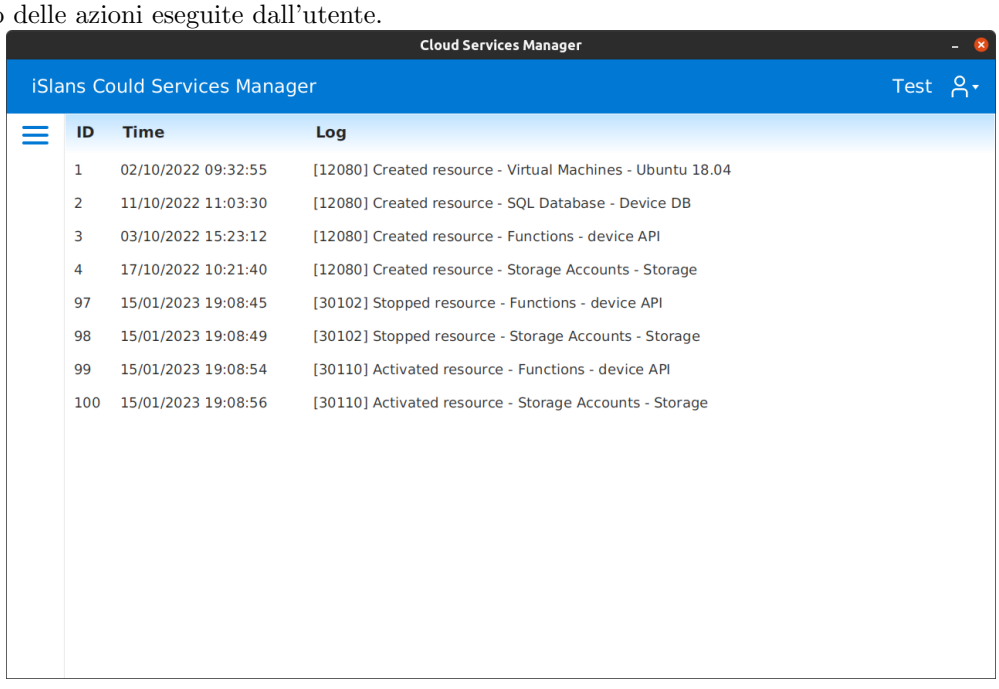
2.1.4 All Services

Sono mostrati tutti i servizi offerti attualmente dalla piattaforma Cloud. Dal menù a scomparsa, l'utente può attivare un nuovo servizio, apparirà un Dialog con le impostazioni del nuovo servizio da attivare.



2.1.5 Activity Log

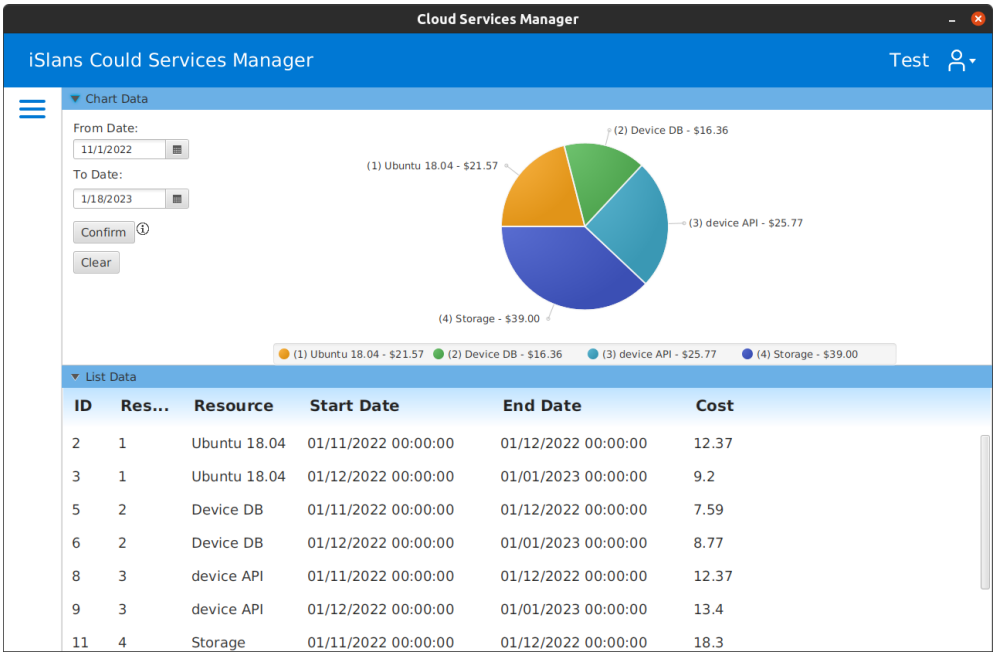
Vi è lo storico delle azioni eseguite dall'utente.



2.1.6 Cost Center

Sono mostrati i costi dei servizi usufruiti. Di default, sono prelevati i dati degli ultimi tre mesi. La pagina ha due sezioni: Char Data e List Data

- In Chart Data, viene mostrato nel grafico i costi sostenuti per servizio, nel periodo di tempo selezionato.
- In List Data sono mostrati i singoli costi.



2.2 Interazione con Server

Per le richieste http nell'applicazione, è stato usato la libreria Java Http Client (java.net.http) Vi è una classe `Request.java` [1] che utilizza Http Client ed espone i metodi utilizzati dall'applicazione per la comunicare con il Server.

Le risposte del Server sono in formato json. Sono decodificate, a seconda delle situazioni, con deserializzazione in classi specifiche o in JsonObject generico.

Se le richieste http non vanno a buon fine (e.g. Server non disponibile), comparirà un relativo Alert Dialog per avvisare l'utente.

2.3 Struttura e scelte progettuali

Dato il numero non basso di file nel progetto, i file java sono organizzati in più sottocartelle (subpackage)

```
Applicazione
└── src/main/java/
    └── [package]/
        ├── http/ ..... Http requests to Server
        ├── exception/ ..... Custom exceptions
        ├── manager/ ..... Application main files
        │   ├── ...
        │   ├── component/ ..... Application components
        │   └── beans/ ..... JavaBeans
        └── App.java
```

Una scelta progettuale di interesse può essere riguardo le diverse schermate dell'applicazione mostrate in precedenza. Questi hanno due componenti in comune tra loro, TopBar e SideBar; mentre il MainBody cambia in base alla schermata. Questa struttura modulare si riflette nei file fxml ed i relativi Controller. Vi è un manager.fxml [2] come schermata di base, che include i componenti fxml necessari, caricati staticamente fin dall'inizio o dinamicamente durante il funzionamento.

[1] `src/main/java/progetto/applicazione/http/Request.java`
[2] `src/main/resources/progetto/applicazione/manager/manager.fxml`

3 Server

Il Server si avvia solo se il database è stato creato.
Le API esposte sono con con prefisso `/api/v1`
Le risposte delle API sono in formato json

3.1 Sessione e Token

Dopo la prima autenticazione con username e password, il server genera e restituisce un Token. Questo sarà necessario al client per accedere alle API che interagiscono sulle risorse dell'utente.
Il Token generato ha una scadenza, dopo la quale non sarà più valido.
Se un API riceve un Token non valido, risponderà con lo stato 401 Unauthorized.

3.2 Spring JPA

Per usare Spring JPA, sono state mappate le principali tabelle del database in Entities [3] ed usato i JPA Repository [4]
Per le query semplici sono utilizzate i derived Query di Spring JPA. Mentre quelle non immediate con join di tabelle, per semplicità sono create usando nativeQuery.

3.3 Lista API

Riporto qui brevemente la lista delle API esposte, senza entrare nei dettagli sui dati in input richiesti e le risposte date.

```
GET  /user/costs
GET  /user/activity
POST /user/login

GET  /resource/user
POST /resource/create
POST /resource/activate
POST /resource/stop
DELETE /resource

GET  /subscription/user
POST /subscription/activate
POST /subscription/stop

GET  /service/all
GET  /service/plans
```

[3] `src/main/java/progetto/server/database/table/...`
[4] `src/main/java/progetto/server/database/...`