Università degli studi di Salerno

Ingegneria Informatica

Laurea Magistrale

Corso di

Mobile Programming

Documentazione Homework: *A CHAT*

The CW Chat

Gruppo 2

Wael Karman	0622700787
Christian Gambardella	0622700850

1 INDICE

1. INTRODUZIONE 3

2. FUNZIONAMENTO 4

2.1 LATO SERVER 4

2.2 LATO CLIENT 6

3. PRINCIPALI FUNZIONI 9

3.1 CLIENT 9

3.2 SERVER 9

3.3 PROTOCOLLO DI SCAMBIO 9

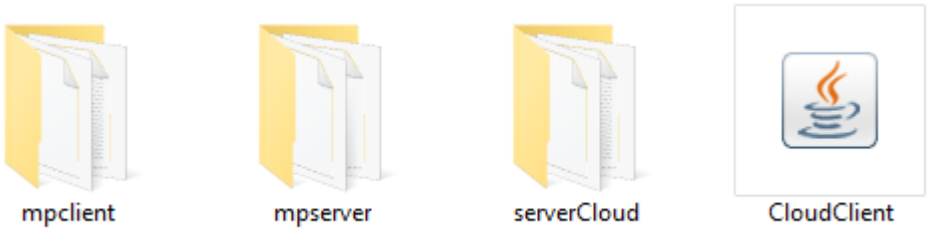
4. ARCHITTETURA 10

5. OVERVIEW LATO CLOUD 11

1. INTRODUZIONE

Si presenta il seguente elaborato per documentare il funzionamento di un **applicativo distribuito scalabile nel numero di utenti**, sviluppato nel corso di *Mobile Programming*, che permette lo scambio di messaggi **fra utenti dislocati sulla rete**.

La consegna effettuata comprende i seguenti file:



Locale

- *mpclient* : contiene il progetto relativo al lato client per permettere agli utenti l'accesso alla chat.
- *mpserver* : contiene il pogetto relativo al lato server per gestire la comunicazione tra i client e quindi le connessioni multiple (una per client).

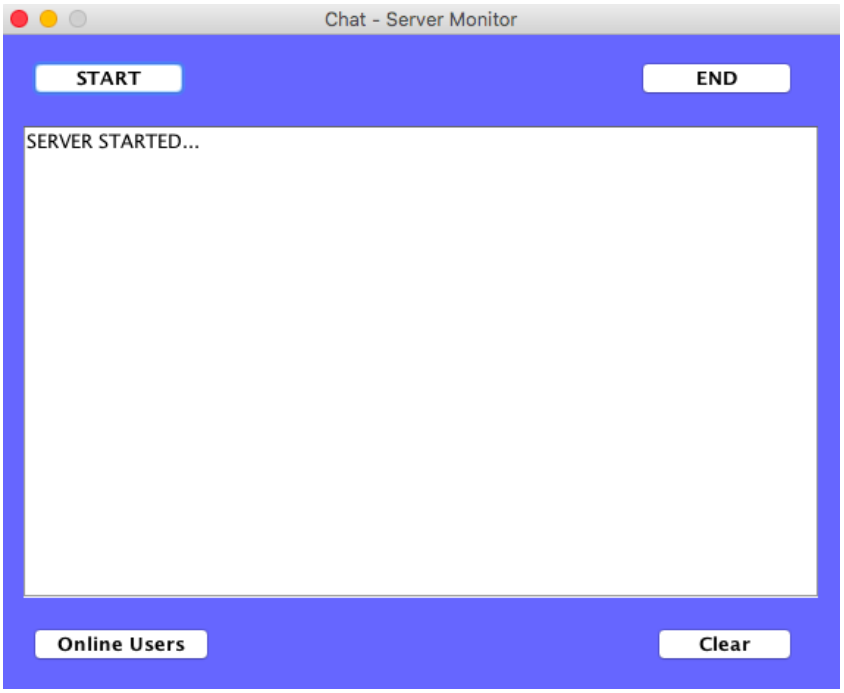
Distribuito – Cloud

- *serverCloud* : contiene l'applicativo lato server lanciato su *Cloud Platform* per consentire una comunicazione **end-to-end** scalabile sulla rete.
- *CloudClient* : file di formato .jar per consentire al client l'accesso diretto alla chat.

2. FUNZIONAMENTO

Nel seguente capitolo viene mostrato il funzionamento dell'applicativo tramite le diverse interfacce.

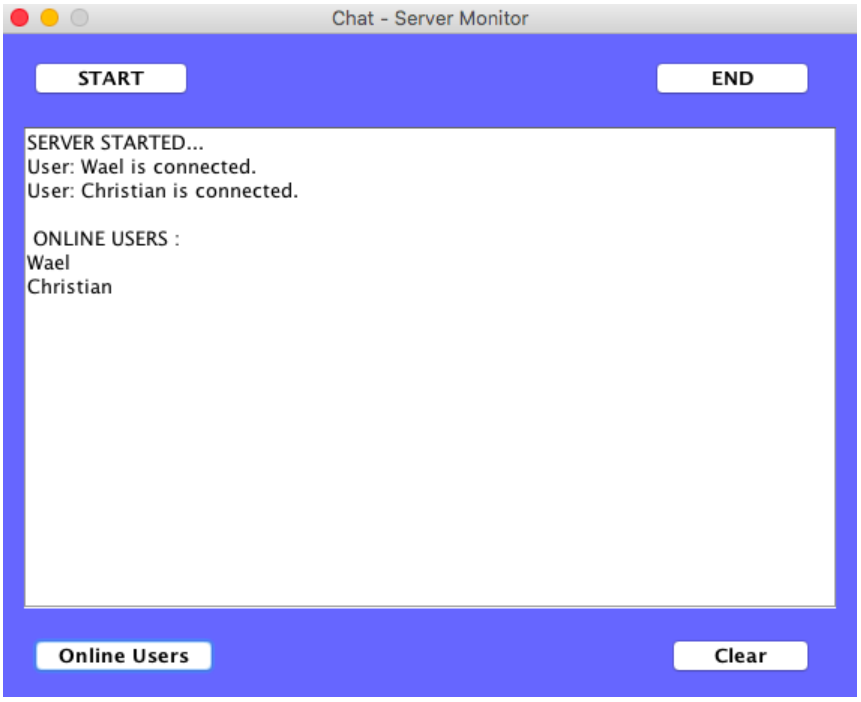
2.1 LATO SERVER



Qui viene data visione della schermata principale del server, dove è possibile eseguire lo stesso tramite il pulsante ***START***.
Se risulta essere correttamente attivo viene mostrato il messaggio riportato nella TextArea.

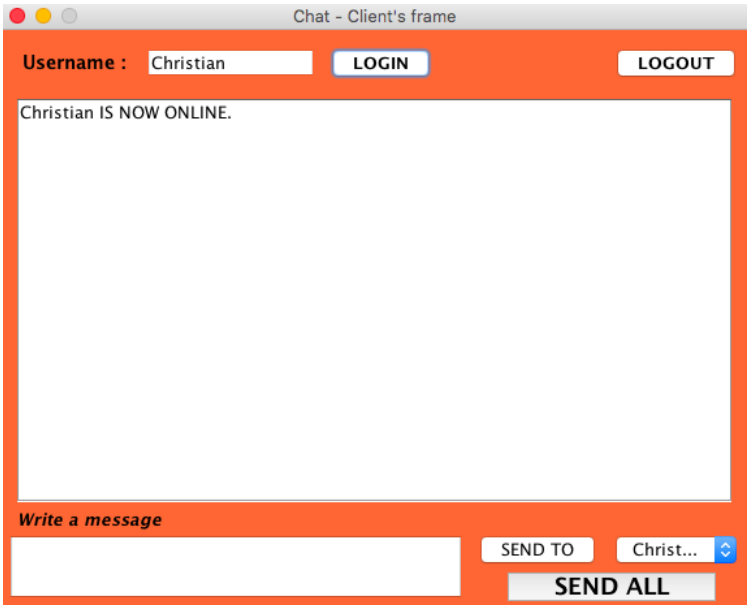
In alto a sinistra è presente invece il pulsante ***END*** che permette la chiusura dell'esecuzione del Server in seguito alla ricezione della risposta affermativa data al messaggio di selezione, come mostrato in seguito.

Inoltre sono presenti altri due pulsanti, ***Clear*** permette la pulizia, e quindi la cancellazione, del testo all'interno della textArea e ***Online Users*** invece la visualizzazione degli utenti attualmente connessi (figura in seguito).



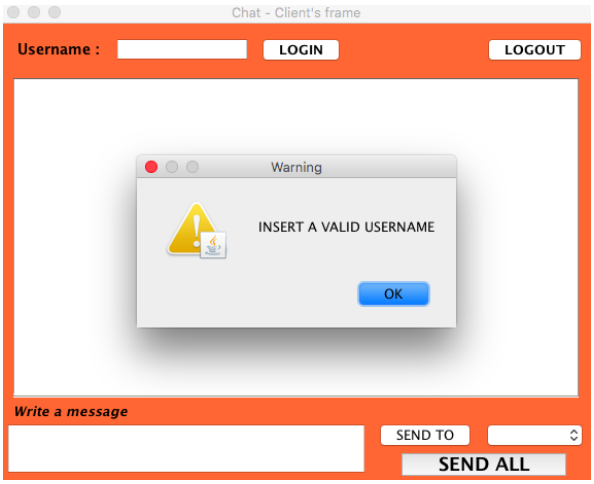
2.2 LATO CLIENT

La schermata principale lato client è la seguente. In alto è presenta una TextField dove è possibile inserire la propria username con la quale collegarsi alla chat; questo viene fatto in seguito all'azionamento del pulsante ***LOGIN***. Se non ci sono particolari errori (mostrati successivamente) l'utente accede correttamente alla chat e gli viene mostrato il messaggio relativo.
Inoltre ogni utente è identificato da un'unica username quindi in caso di username uguale da un utente già connesso, la connessione viene semplicemente rifiutata.

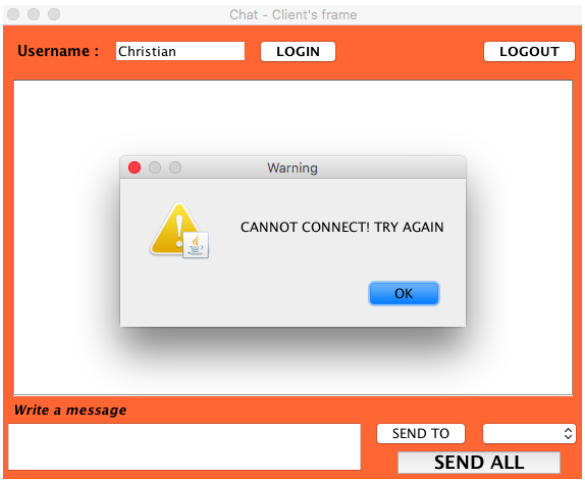


Gli errori che l'utente potrebbe ricevere nell'accesso alla chat potrebbero essere i seguenti:

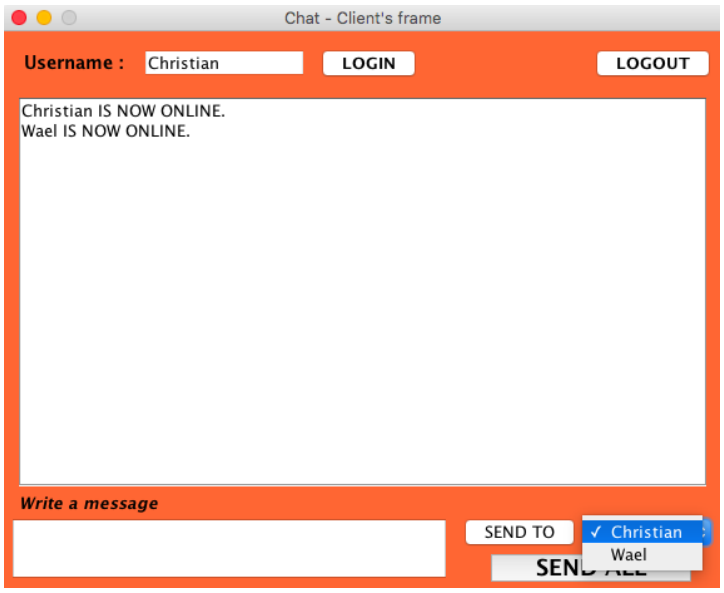
Campo username vuoto



Server non attivo



In seguito alla connessione di un altro utente all'interno della chat, compare una relativa comunicazione e inoltre egli viene aggiunto alla lista degli utenti attualmente attivi. Nell'esempio sottostante, Wael ha appena fatto accesso alla chat.



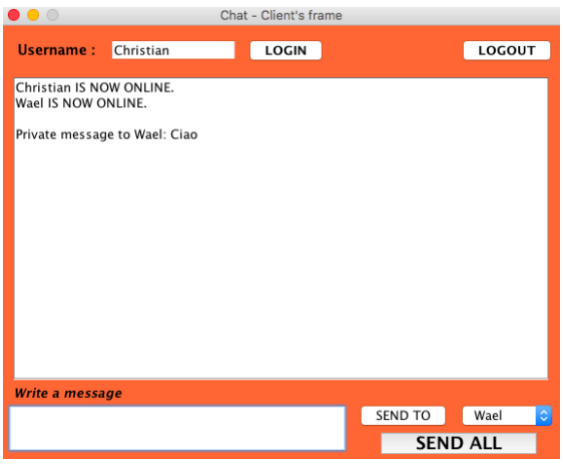
La comunicazione tra diversi client può avvenire in due diverse modalità:

- **One-to-one** permette lo scambio di messaggio tra due singoli utenti

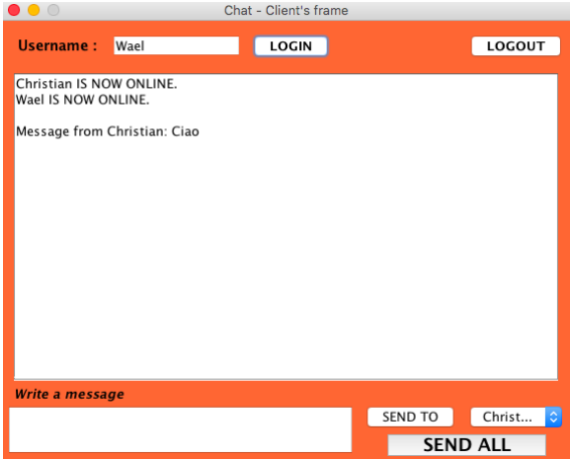
E' possibile inviare un messaggio (che viene scritto nell'apposita textArea in basso nell'interfaccia) a un singolo utente in modalità privata tramite il pulsante ***SEND TO*** in seguito alla selezione del destinatario scelto nella lista adiacente al pulsante stesso.

In seguito vengono mostrati i messaggi che compariranno nell'apposita area di testo ai due utenti coinvolti.

Mittente



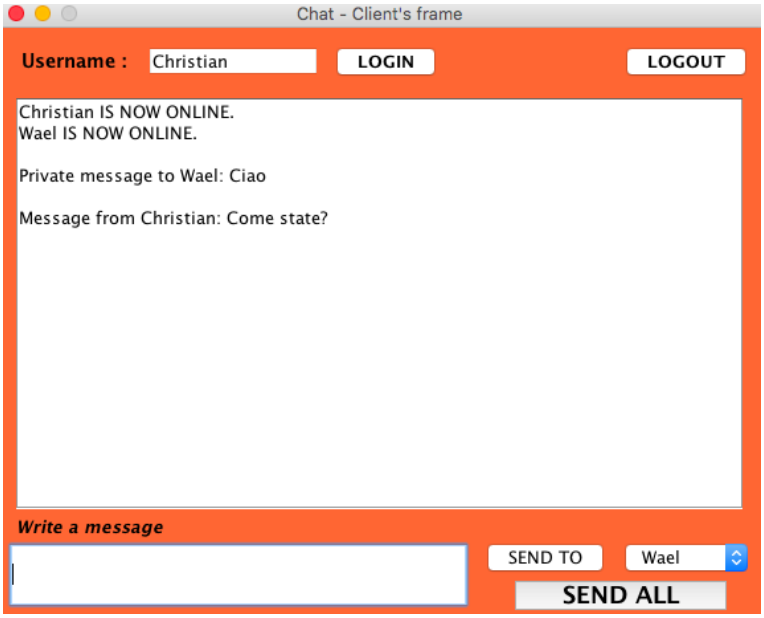
Destinatario



- **Broadcast** permette lo scambio di messaggi tra tutti gli utenti

E' possibile inviare un messaggio (che viene scritto nell'apposita textArea in basso nell'interfaccia) a tutti gli utenti in modalità pubblica tramite il pulsante ***SEND ALL***.

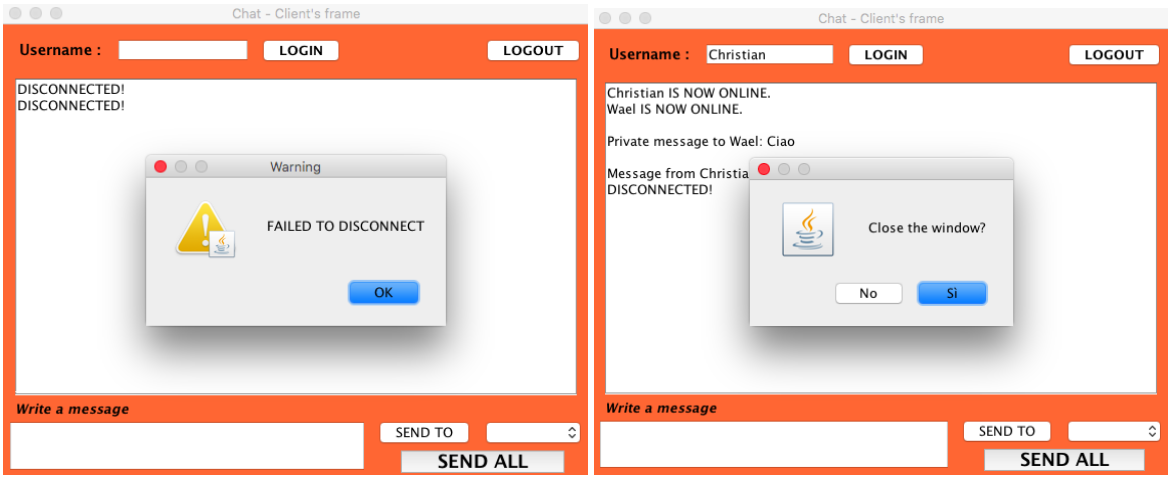
Il seguente messaggio comparirà nell'apposita area di testo a tutti gli utenti connessi.



Infine vengono mostrate altre due situazioni nelle figure sottostanti.

A sinistra è possibile vedere un messaggio di errore che compare nel caso in cui un utente prova a premere ***LOGOUT*** senza aver però prima fatto il login o nel caso in cui il server non è attivo.

A destra invece la schermata relativa alla disconnessione dell'utente dalla chat in seguito all'azionamento del pulsante ***LOGOUT***.



3. PRINCIPALI FUNZIONI

3.1 CLIENT

```
public class IncomingReader implements Runnable
```

La classe **IncomingReader** implementa *Runnable* e istanzia un *Thread* che gestisce la comunicazione con il server secondo il **protocollo di scambio** definito in questo documento.

3.2 SERVER

```
public class ServerStart implements Runnable
```

La classe **ServerStart** implementa *Runnable* e istanzia un *Thread* **A** per la gestione delle connessioni in ingresso. Il *Thread A* ascolta il canale attendendo una nuova connessione (con il metodo *accept()*). All' occorrenza di una connessione, il *Thread* **A** crea un altro *Thread **B*** affidandogli la comunicazione con l'utente **X1** appena connesso e ritorna in ascolto nell'attesa di una nuova connessione.

```
public class ClientHandler implements Runnable
```

La classe **ClientHandler** implementa *Runnable* e prevede il lancio del *Thread **B*** (citato nella descrizione della classe *ServerStart*) che gestisce la comunicazione con un singolo utente. (dunque ad ogni nuovo client un nuovo *Thread **Bx*** viene lanciato) *ClientHandler* gestisce dunque lo scambio di messaggi con l'utente **Xn** secondo un fissato **protocollo di comunicazione.**

3.3 PROTOCOLLO DI SCAMBIO

La comunicazione avviene con l'invio dei messaggi nelle tabelle separati dal separatore "**:**"

- Esempio *Server to Client* riga 1 **->** "**wael : - : connect**"
- Esempio *Client to Server* riga 3 **->** "**wael:Ciao come stai?:chat:christian**"

Server to Client

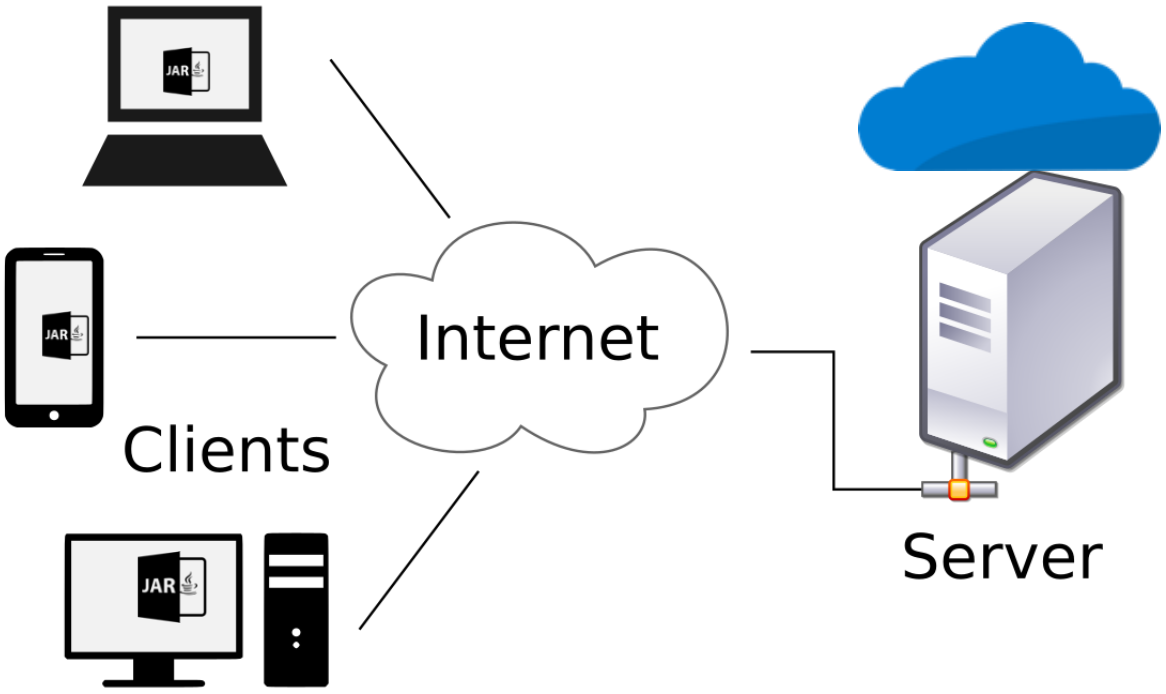
Sender or Users	Message	Message Type
<user >	-	*connect*
<user>	*Disconnect or <reason>*	*disconnect*
<sender>	*<message>*	*broadcast*
<users> connected yet	*To Update*	*update*

Client to Server

Username sender	Message	Message Type	Destination
<username>	-	*connect*	*broadcast*
<username>	*Disconnect or <reason>*	*disconnect*	*broadcast*
<username>	*<message>*	*chat*	*<user>/broadcast*

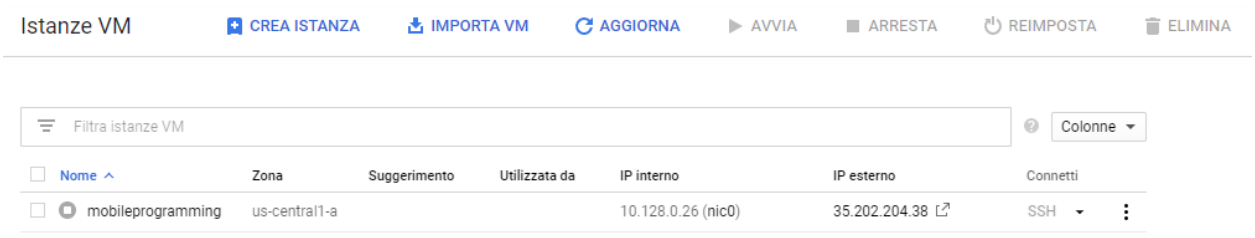
4. ARCHITTETURA

L'architettura utilizzata per l'applicativa è **Client-Server** nella quale un terminale (client) si connette a un server per la fruizione di servizi appoggiandosi ad un'architettura protocollare.

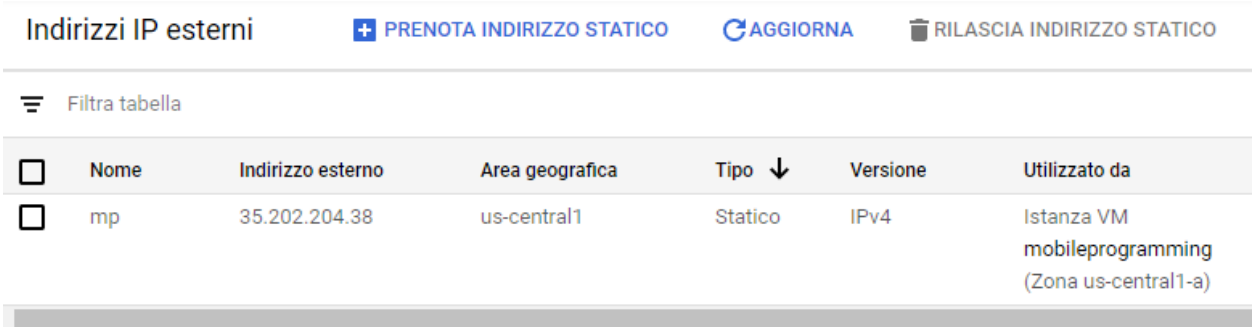


5. OVERVIEW LATO CLOUD

L' applicativo è eseguito su **server linux** creato su *Cloud Platform*



per la comunicazione **end to end** è necessario assegnare alla VM un indirizzo **IP Statico**



oltre a consentire le connessioni ingresso/uscita su una porta scelta (**porta 33333**)