

QB SOFTWARE



×



UNIVERSITÀ DEGLI STUDI DI PADOVA

CORSO DI INGEGNERIA DEL SOFTWARE

ANNO ACCADEMICO 2023/2024

---

# Manuale Utente

---

Contatti: [qbsoftware.swe@gmail.com](mailto:qbsoftware.swe@gmail.com)



## Registro delle modifiche

V.	Data	Membro	Ruolo	Descrizione
1.0.0	05/05/2024	A. Domuta	Responsabile	Approvazione documento
0.6.0	28/04/2024	A. Bustreo	Verificatore	Controllo qualità
	27/04/2024	S. Rovea	Programmatore	Effettuata revisione
0.5.0	26/04/2024	S. Rovea	Verificatore	Controllo qualità
	26/04/2024	A. Bustreo	Programmatore	Aggiornato la sezione relativa agli stress test
0.4.0	24/04/2024	R. Fontana	Verificatore	Controllo qualità
	24/04/2024	S. Rovea	Programmatore	Redatto la sezione relativa a Postman
0.3.0	12/04/2024	A. Giurisato	Verificatore	Controllo qualità
	11/04/2024	A. Feltrin	Programmatore	Redatto la sezione di supporto e dei requisiti
0.2.0	10/04/2024	A. Domuta	Verificatore	Controllo qualità
	10/04/2024	A. Bustreo	Programmatore	Redatto una prima versione della sezione di stress test
0.1.0	05/04/2024	S. Rovea	Verificatore	Controllo qualità
	04/04/2024	A. Bustreo	Programmatore	Redatto la sezione di avvio e di introduzione



## Indice

<b>1</b>	<b>Introduzione</b>	<b>4</b>
1.1	Scopo del documento . . . . .	4
1.2	Scopo del prodotto . . . . .	4
1.3	Glossario . . . . .	4
1.4	Riferimenti . . . . .	4
1.4.1	Normativi . . . . .	4
1.4.2	Informativi . . . . .	5
<b>2</b>	<b>Requisiti di sistema</b>	<b>6</b>
2.1	Requisiti tecnici . . . . .	6
<b>3</b>	<b>Avvio</b>	<b>8</b>
3.1	Avvio client . . . . .	8
3.2	Configurazione client e collegamento al server . . . . .	9
3.3	Spegnimento . . . . .	10
<b>4</b>	<b>Postman</b>	<b>11</b>
4.1	Introduzione a Postman . . . . .	11
<b>5</b>	<b>Stress test</b>	<b>12</b>
5.1	Locust . . . . .	12
5.2	Descrizione dei test . . . . .	12
5.2.1	testEmails . . . . .	12
5.2.2	testIdentities . . . . .	12
5.3	Esecuzione dei test . . . . .	13
5.4	Configurazione parametri . . . . .	13
5.5	Visualizzazione dei risultati . . . . .	14
<b>6</b>	<b>Supporto tecnico</b>	<b>16</b>



## Elenco delle figure

1	Schermata di inserimento email. . . . .	9
2	Schermata di inserimento URL di connessione. . . . .	10
3	Schermata di configurazione dei parametri. . . . .	14
4	Schermata monitoraggio statistiche. . . . .	14
5	Schermata monitoraggio grafici. . . . .	15



# 1 Introduzione

## 1.1 Scopo del documento

Il gruppo **QB Software**, con il presente documento intende spiegare come utilizzare il *prodotto<sub>G</sub>* sviluppato. In particolare contiene istruzioni su come avviare il *server<sub>G</sub>* di posta elettronica e come avviare e collegarsi al *client<sub>G</sub>* scelto per poterlo utilizzare. Fornisce anche una guida all'installazione di *Postman<sub>G</sub>* che permetterà di verificare la funzionalità del prodotto, nonché di eseguire gli *stress test<sub>G</sub>* richiesti dal *committente<sub>G</sub>*.

## 1.2 Scopo del prodotto

Lo scopo del prodotto è principalmente esplorativo. L'obiettivo è quello di capire se implementare o meno il *protocollo<sub>G</sub>* di posta elettronica JMAP nel loro attuale sistema *Carbonio<sub>G</sub>*, che è un sistema di collaborazione on-line orientato all'*efficienza<sub>G</sub>* dell'organizzazione del team. Dato che JMAP è stato ideato con l'obiettivo di semplificare ulteriormente la comunicazione tra client e server, è *ragionevole<sub>G</sub>* valutare se l'integrazione tra i due migliori l'efficienza e la *performance<sub>G</sub>* generale di Carbonio.

## 1.3 Glossario

Al fine di una maggiore chiarezza dei contenuti redatti in questo documento, viene fornito in allegato il *Glossario v2.0.0*, dove vengono definiti tutti i termini con un significato particolare o di rilievo nell'ambito del progetto. Un termine presente nel *Glossario* viene contrassegnato dal testo formattato in corsivo, seguito dalla lettera "G" a pedice.

## 1.4 Riferimenti

### 1.4.1 Normativi

- *Norme di Progetto v2.0.0*;
- Lezione PD2- Regolamento del progetto didattico: <https://www.math.unipd.it/~tullio/IS-1/2023/Dispense/PD2.pdf>  
[Online - PDF; ultima visita 21/12/2023]
- Capitolato d'appalto C8:



- <https://www.math.unipd.it/~tullio/IS-1/2023/Progetto/C8.pdf>  
[Online - PDF; ultima visita 21/12/2023];
- <https://www.math.unipd.it/~tullio/IS-1/2023/Progetto/C8p.pdf>  
[Online - PDF; ultima visita 21/12/2023].

#### 1.4.2 Informativi

- *Glossario v2.0.0;*
- *Piano di Qualifica v2.0.0;*



## 2 Requisiti di sistema

Per poter eseguire il prodotto è necessario che il *dispositivo<sub>G</sub>* utilizzato soddisfi i *requisiti<sub>G</sub>* descritti in seguito.

### 2.1 Requisiti tecnici

Il prodotto è stato sviluppato utilizzando determinate tecnologie pertanto è necessario che siano installate anche all'interno del *sistema<sub>G</sub>* che verrà utilizzato per poter procedere all'installazione e al conseguente avvio del server, quali:

- E' necessario che sia installata una *versione<sub>G</sub>* di *Java<sub>G</sub>* non precedente alla 21 LTS, la cui *documentazione<sub>G</sub>* e guida per l'installazione si trova qui: <https://www.oracle.com/it/java/technologies/downloads/#java21>;
- il dispositivo deve contenere *Docker<sub>G</sub>* come sistema per la gestione container, la guida per l'installazione fornita di seguito: <https://docs.docker.com/engine/install/>;
- il nostro sistema di *build<sub>G</sub>* è *Maven (mvn)<sub>G</sub>* per cui è necessaria la versione più recente compatibile con quella di Java, introduzione e guida per l'installazione è consultabile qui: <https://maven.apache.org/what-is-maven.html>;
- per la gestione del *Database<sub>G</sub>* abbiamo scelto di adottare *MongoDB<sub>G</sub>*, la cui documentazione e installazione sono consultabili qui: <https://www.mongodb.com/docs/manual/installation/>.

Per quanto riguarda invece il client, necessario per il collegamento al server, si descrivono in seguito i requisiti per il corretto funzionamento:

- è necessario utilizzare un client JMAP che permetta di collegarsi utilizzando *HTTP<sub>G</sub>*, e non *HTTPS<sub>G</sub>*. Il team ha optato per *Ltt.rs<sub>G</sub>* per Android le cui informazioni si trovano qui: <https://codeberg.org/iNPUTmice/lttrs-android>;
- per eseguire il client si è deciso di utilizzare *Android Studio<sub>G</sub>*, scaricabile qui:

<https://developer.android.com/codelabs/basic-android-kotlin-compose-install-android-studio#0>.



Rimane dunque l'accesso alla *repository<sub>G</sub>* contenente il codice sorgente del server:

<https://github.com/QB-Software-swe/PoC>.

Per l'importazione delle richieste, l'esecuzione dei test<sub>G</sub> e la loro interpretazione si è deciso di utilizzare l'applicazione Postman. È possibile scaricarla per il proprio sistema operativo dal seguente link: <https://www.postman.com> e sarà necessario creare un account.

Per eseguire gli stress test si userà Locust<sub>G</sub>, un framework che permetterà di simulare un gran numero di interazioni simultanee sulla nostra applicazione per valutarne la performance. Essendo scritto in Python<sub>G</sub> sarà necessario averlo installato sul proprio dispositivo, dal link: <https://www.python.org/downloads/>. È possibile scaricarlo dal seguente link: <https://locust.io> dove sono presenti anche le istruzioni per l'installazione.





### 3 Avvio

Per l'avvio dell'applicazione è necessario verificare che tutti i requisiti siano soddisfatti. Per verificarlo aprire una finestra del terminale e in base al proprio sistema operativo verificare che le versioni giuste dei requisiti siano state installate, come specificato in seguito:

```
$ docker version
$ java -version
$ mvn -version
```

Assicurarsi di aver avviato Docker e per eseguire l'applicazione da container usare il seguente comando:

```
$ docker compose up --build
```

Per fare invece la build del prodotto ed eseguirlo senza container si utilizzano i seguenti comandi:

```
$ mvn clean package
$ java -jar target/poc_jmap-0.1-SNAPSHOT-shaded.jar
```

Per la configurazione del server utilizzare la porta 9999. Gli endpoints sono:

```
/.well-known/jmap
/api
/download
/upload
```

Mentre l'URL del server è: [http://<server\\_ip>:9999/.well-known/jmap](http://<server_ip>:9999/.well-known/jmap)

#### 3.1 Avvio client

Se si desidera scegliere un client diverso da quello proposto nella



sezione 2, consultare il seguente link per verificare quelli compatibili:

<https://jmap.io/software.html>. Altrimenti a questo punto si dovrebbe aver installato sul proprio dispositivo il client proposto.

Avviare dunque Android Studio per il client scelto Ltt.rs e, dopo aver modificato l'indirizzo IP nel file `network_security_config.xml` in modo che corrisponda all'indirizzo localhost, utilizzare l'emulatore integrato per visualizzare il prodotto in funzione.

### 3.2 Configurazione client e collegamento al server

Appena avviato il client sarà necessario inserire l'email con la quale si intende accedere.

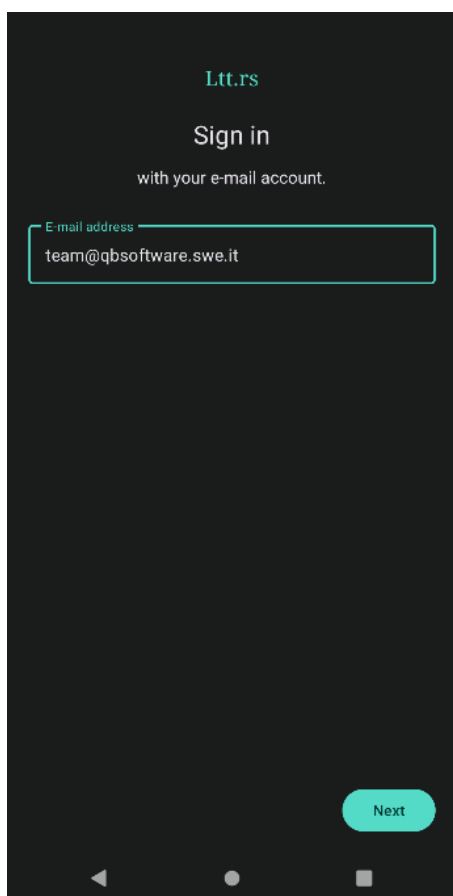


Figura 1: Schermata di inserimento email.

Successivamente, inserire l'URL del server e la porta, come mostrato in figura.

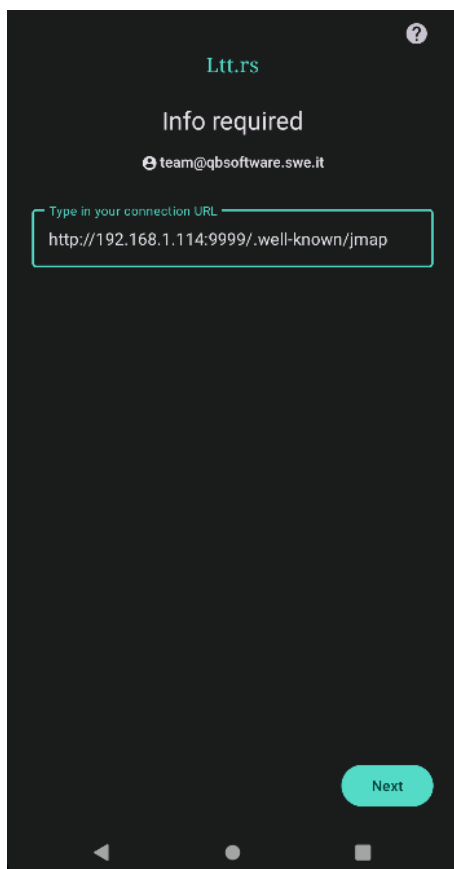


Figura 2: Schermata di inserimento URL di connessione.

### 3.3 Spegnimento

Per lo spegnimento invece si può usare il seguente comando:

```
$ docker compose down
```



## 4 Postman

### 4.1 Introduzione a Postman

Postman è un'applicazione software utilizzata per testare, sviluppare e documentare le *API*. Tra le sue principali caratteristiche, vi sono la capacità di creare e organizzare collezioni di richieste, l'automazione dei test delle API, la gestione delle variabili per facilitare la personalizzazione delle richieste e la generazione di documentazione automatica delle API.

Lo scopo di questa sezione è utilizzare Postman per facilitare il processo di importazione delle richieste e la loro visualizzazione nel dettaglio. Permette anche l'esecuzione dei test e l'interpretazione delle relative risposte. Per l'installazione le istruzioni sono presenti nella sezione [2](#).



## 5 Stress test

### 5.1 Locust

Locust è un framework open-source utilizzato per testare le prestazioni delle applicazioni web. È scritto in Python e consente agli sviluppatori di simulare il comportamento di un gran numero di utenti simultanei che interagiscono con un'applicazione. Utilizzando Locust, è possibile definire scenari di test personalizzati e offre anche strumenti per monitorare e analizzare le prestazioni del sistema durante i test, consentendo agli sviluppatori di identificare e risolvere eventuali problemi.

Lo scopo di questa sezione è utilizzare Locust per permetterci di eseguire degli stress test soddisfacenti e verificare la performance della nostra applicazione, simulando appunto un gran numero di operazioni successive. Le istruzioni per l'installazione sono presenti nella sezione [2](#).

### 5.2 Descrizione dei test

#### 5.2.1 testEmails

In questo test di carico, si simula l'invio di un numero elevato di email da parte di un utente. Ecco cosa fa nel dettaglio:

- **Autenticazione:** All'inizio del test, l'utente si autentica sul server di posta;
- **Generazione delle caselle di posta:** Dopo l'autenticazione, il test genera delle caselle di posta sul server;
- **Creazione della casella di posta "sent":** Controlla se esiste una casella di posta per le email inviate "sent". Se non esiste, la crea;
- **Generazione di email:** Infine, il test simula l'invio di email da molteplici utenti. Ogni email viene generata con un indirizzo email del mittente casuale, scelto casualmente da una pool predefinita.

#### 5.2.2 testIdentities

In questo test di carico si simula la creazione di un numero elevato di identità. Ecco cosa fa nel dettaglio:



- **Autenticazione:** All'inizio del test, l'utente si autentica sul server di posta;
- **Ottenimento dell'ID dell'account:** Dopo l'autenticazione, il test ottiene l'ID dell'account dell'utente;
- **Generazione di identità:** Il test consiste in una task che genera una nuova identità. Questa identità ha un nome casuale scelto da un pool predefinito namePool e un indirizzo email casuale scelto da un altro pool predefinito emailPool.

### 5.3 Esecuzione dei test

Per eseguire correttamente i test bisogna seguire i seguenti passaggi:

- **Avviare il server di posta:** Prima di eseguire i test, è necessario avviare il server di posta. Per farlo, aprire un terminale e spostarsi nella cartella del progetto. Eseguire il comando: `docker-compose up --build;`
- **Avviare il test:** Aprire un altro terminale e spostarsi nella cartella del progetto. Eseguire il comando:  

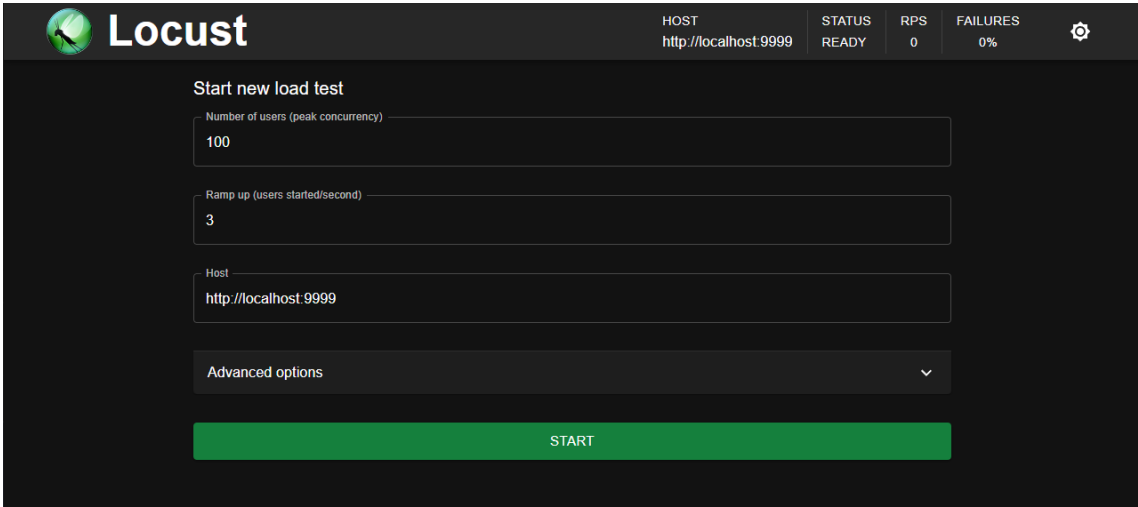
```
python -m locust -f .\locust\test.py --host=http://localhost:9999.
```

 Bisogna sostituire `test.py` con il nome del test che si vuole eseguire e anche modificare l'host se il server di posta è in esecuzione su un altro indirizzo.
- **Visualizzazione interfaccia utente:** Aprire un browser e andare all'indirizzo `http://localhost:8089`. Qui è possibile visualizzare i risultati dei test in tempo reale utilizzando l'interfaccia utente di locust.

### 5.4 Configurazione parametri

Tramite l'interfaccia utente di Locust sarà necessario impostare i seguenti parametri:

- **Number of users (peak concurrency):** Numero di utenti massimo che si vuole simulare durante il test;
- **Ramp up:** Definisce il tasso di aggiunta di nuovi utenti simulati nel tempo. Ad esempio, impostando il ramp up a 3 utenti al secondo e il numero di utenti a 100, Locust inizierà con 3 utenti e ne aggiungerà altri 3 ogni secondo fino al raggiungimento del numero massimo di 100 utenti.

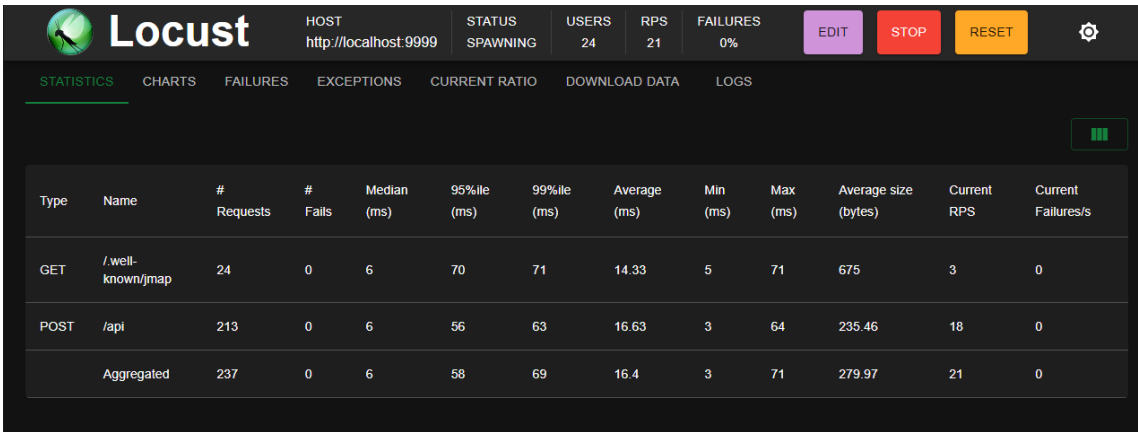


The image shows the Locust configuration interface. At the top, there's a header with the Locust logo and a status bar showing HOST (http://localhost:9999), STATUS (READY), RPS (0), and FAILURES (0%). Below the header, there's a section titled "Start new load test". It contains three input fields: "Number of users (peak concurrency)" with the value 100, "Ramp up (users started/second)" with the value 3, and "Host" with the value http://localhost:9999. There's also an "Advanced options" dropdown menu. At the bottom of this section is a large green "START" button.

Figura 3: Schermata di configurazione dei parametri.

## 5.5 Visualizzazione dei risultati

Una volta impostati i parametri, è possibile avviare il test premendo il pulsante "START". Durante l'esecuzione del test, è possibile visualizzare i risultati in tempo reale nell'interfaccia utente di Locust. Una volta terminato il test, è possibile visualizzare i risultati completi e le statistiche dettagliate.



The image shows the Locust statistics monitoring interface. At the top, there's a header with the Locust logo and a status bar showing HOST (http://localhost:9999), STATUS (SPAWNING), USERS (24), RPS (21), and FAILURES (0%). There are also buttons for EDIT, STOP, and RESET. Below the header, there's a navigation bar with tabs: STATISTICS, CHARTS, FAILURES, EXCEPTIONS, CURRENT RATIO, DOWNLOAD DATA, and LOGS. The STATISTICS tab is selected. Below the navigation bar, there's a table with the following data:

Type	Name	# Requests	# Fails	Median (ms)	95%ile (ms)	99%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
GET	/well-known/jmap	24	0	6	70	71	14.33	5	71	675	3	0
POST	/api	213	0	6	56	63	16.63	3	64	235.46	18	0
Aggregated		237	0	6	58	69	16.4	3	71	279.97	21	0

Figura 4: Schermata monitoraggio statistiche.



Figura 5: Schermata monitoraggio grafici.





## 6 Supporto tecnico

Per qualsiasi dubbio o problema contattare il nostro team all'e-mail [qbsoftware.swe@gmail.com](mailto:qbsoftware.swe@gmail.com) assicurandosi di aver fornito una accurata descrizione del problema. In caso questo non sia risolvibile tramite e-mail è possibile programmare un meeting in cui guidere-mo passo passo l'utente all'avvio della nostra applicazione e alla risoluzione di eventuali problemi.