1. SERVER

- 1.1. Configurazione custom server
- 1.2. Configurazione spring server
- 1.3. Eseguire default server
- 1.4. Eseguire info server
- 1.5. Creare ed eseguire il file .jar

2. WEB CLIENT

- 2.1. Installazione dashboard
- 2.2. Configurazione della dashboard
- 2.3. Avvio dashboard in locale
- 2.4. Build & deploy dashboard
 - 2.4.1. Shortcut npm

I titoli di intestazione indicano le directory del progetto.

1. SERVER

1.1. Configurazione custom server

Il file di configurazione del custom server si trova seguendo il path: /src/main/java/com/server/CustomServer/Configuration/config.json

All'interno del file JSON sono presenti i seguenti parametri di configurazione:

- PORT: porta di ascolto del custom server
- CORE_POOL_SIZE: numero minimo di thread del thread pool impiegato per i thread worker
- MAX_POOL_SIZE: numero massimo di thread del thread pool impiegato per i thread worker
- **KEEPALIVETIME:** secondi per cui un thread del thread pool, a cui non viene sottoposto un task, viene "tenuto in vita" prima di essere eliminato
- **TIMEOUT_SELECTOR:** millisecondi per cui il selettore multiplexer rimane bloccato qualora non ci fossero canali da cui leggere o in cui scrivere
- QUEUE_SIZE: lunghezza della coda dei task per i thread worker del thread pool

1.2. Configurazione spring server

Il file di configurazione del custom server si trova seguendo il path: /src/main/resources/application.properties

All'interno del file sono presenti i seguenti parametri di configurazione:

- server.port: porta di ascolto del server spring.
- server.error.path: parametro necessario per sfruttare il routing di react.

- **spring.banner.location:** parametro che serve a personalizzare il banner di avvio del server (WindFlow Dashoard in ascii nel nostro caso).
- **spirng.output.ansi.enabled:** parametro necessario per far stampare il banner personalizzato.
- **logging.level.com.server.CustomServer.ServerWF**: indica il livello di logging per il custom server.

Di default è settato a INFO, mostrando perciò solo le informazioni basilari. Settandolo col flag DEBUG verranno mostrate tutte le operazioni eseguite dal custom server sulla socket destinata alle applicazioni WindFlow (accettazione di nuovi client, read, write etc).

1.3. Eseguire default server

Facendo eseguire lo script run_server verrà avviato il server in modalità INFO

1.4. Eseguire info server

Facendo eseguire lo script run_info_server verrà avviato il server in modalità DEBUG. In questa modalità verranno mostrate tutte le operazioni eseguite dal custom server sulla socket destinata alle applicazioni WindFlow (accettazione di nuovi client, read, write etc).

1.5. Creare ed eseguire il file .jar

Per creare il file .jar, dalla directory Server, digitare il comando: mvn clean package

Per eseguire il file .jar, dalla directory Server, digitare il comando: java -jar target/DashboardWindflow-1.0-SNAPSHOT.jar

2. WEB_CLIENT

2.1. Installazione dashboard

Dopo essersi recati nella directory web_client, digitare il comando: npm install

2.2. Configurazione della dashboard

Il file di configurazione della web dashboard si trova al seguente path: /src/Configuration/config.json

I parametri di configurazione sono i seguenti:

• **url:** url per connettersi al server spring. Questo parametro è utile solo in fase di sviluppo e permette di testare la dashboard in locale, connettendosi al server locale o

remoto.

Quando si fa la build del progetto questo parametro deve contenere solo le due virgolette (url: "").

• **timeout:** timeout chiamate GET al server. Di default è impostato a 1 secondo. <u>Evitare di cambiare il valore a questo parametro.</u>

2.3. Avvio dashboard in locale

Dopo aver effettuato l'installazione, per avviare la dashboard in locale digitare il comando: npm start

Questa modalità viene usata per lo sviluppo e il testing della dashboard prima che venga fatta la build dell'intero progetto. È possibile cambiare l'url del server a cui collegarsi tramite il file di configurazione.

In questa modalità ogni cambiamento al codice viene riportato in tempo reale a schermo.

2.4. Build & deploy dashboard

N.B. Prima di fare la build del progetto ricordarsi di cambiare il parametro <u>url</u> nel file di configurazione in questo modo: **url**: "".

Per la build del progetto digitare il comando: npm run build

Verrà creata una cartella denominata build.

Per il deploy nella cartella Server, copiare tutti i file della directory build nella directory /Server/src/main/resources/static

2.4.1. Shortcut npm

È possibile creare delle shortcut per delle azioni ripetitive come il deploy. Nel file package.json ad esempio io ho creato una shortcut per il deploy sul server locale e una per il server remoto pianosau.

Per crearne una basta aprire il file, andare nell'oggetto indicizzato dalla key "scripts" e creare la propria.

Ecco come appare con quelle che ho creato:

```
"scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject",
    "local-deploy": "cp -r ./build/*
/home/fausto/GitHub/Tesi/Server/src/main/resources/static/",
```

```
"remote-deploy": "scp -r ./build/*
frasca@131.114.3.249:/home/frasca/Tesi/Server/src/main/resources/static/"
}
```

Per avviarle basta poi digitare il comando:

```
npm run <nome-shortcut>
```

Ad esempio per il deploy remoto io digitavo: npm run remote-deploy