Chatbot

-

Manuale Tecnico

**QUESTA PAGINA È LASCIATA**

**INTENZIONALMENTE BIANCA**

|  |
| --- |
| STATO DEL DOCUMENTO |

|  |  |  |  |
| --- | --- | --- | --- |
| TITOLO DEL DOCUMENTO | | | |
| Chatbot Manuale Tecnico | | | |
| CLASSIFICAZIONE DEL DOCUMENTO | | | |
| USO INTERNO | | | |
| EDIZ. | REV. | DATA | AGGIORNAMENTO |
| 1 | 0 | 10/02/2023 | Prima stesura |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| STATO DI AGGIORNAMENTO | | | |
| PAR | EDIZ. | REVISIONE | MOTIVO DELL'AGGIORNAMENTO |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |
| --- | --- |
| NUMERO TOTALE PAGINE: | 13 |

|  |
| --- |
| LISTA DI DISTRIBUZIONE |

INTERNA A:

ESTERNA A:

**Sommario**

[1. Scopo del documento 6](#_Toc126921699)

[2. Architettura 7](#_Toc126921700)

[3. Botfront 8](#_Toc126921701)

[3.1 Installazione 8](#_Toc126921702)

[3.2 Configurazione 8](#_Toc126921703)

[3.3 Addestramento 8](#_Toc126921704)

[4. Iside 9](#_Toc126921705)

[4.1 Installazione 10](#_Toc126921706)

[4.2 Database Iside 10](#_Toc126921707)

[4.3 Configurazione 10](#_Toc126921708)

[5. nginx 11](#_Toc126921709)

[5.1 Installazione 11](#_Toc126921710)

[6. Convertitore 11](#_Toc126921711)

[6.1 Installazione 11](#_Toc126921712)

[6.2 Esecuzione 11](#_Toc126921713)

[6.3 Istruzioni compilare Excel 12](#_Toc126921714)

[6.4 Esempio Excel 13](#_Toc126921715)

[7. Comandi docker 13](#_Toc126921716)

[7.1 Riavvio containers 13](#_Toc126921717)

[7.2 Logs 13](#_Toc126921718)

[7.2 Rimozione Logs 13](#_Toc126921719)

# Scopo del documento

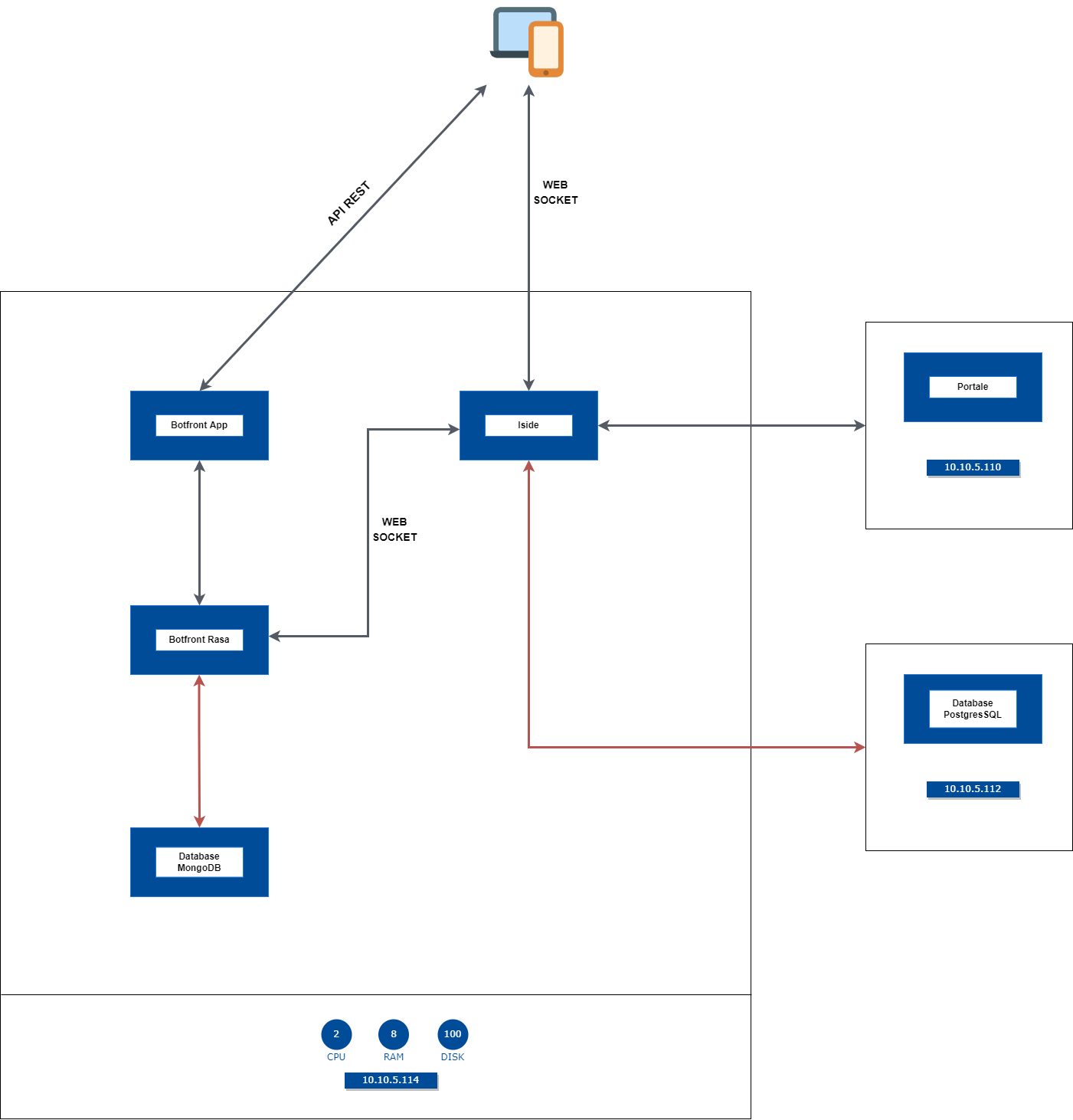
Il documento ha lo scopo di illustrare il manuale tecnico della chatbot commissionato dal Comune di Salerno.

Le componenti che vedremo nel paragrafo “Architettura” sono installate attraverso la tecnologia della virtualizzazione, in particolare con l’utilizzo di Docker.

**Docker** è una piattaforma software che permette di creare, testare e distribuire applicazioni con la massima rapidità. Docker raccoglie il software in unità standardizzate chiamate container che offrono tutto il necessario per la loro corretta esecuzione, incluse librerie, strumenti di sistema, codice e runtime. Con Docker, è possibile distribuire e ricalibrare le risorse per un'applicazione in qualsiasi ambiente, tenendo sempre sotto controllo il codice eseguito.

# Architettura

Nello schema sottostante è rappresentata in grandi linee la comunicazione tra i vari componenti.



# Botfront

Botfront è una piattaforma conversazionale completa costruita su Rasa con interfacce intuitive per creare assistenti virtuali consapevoli del contesto.

Botfront espone tutti i blocchi nativi di Rasa (regole, storie, moduli) in un'interfaccia utente comoda e intuitiva. È importante notare che per supportare le funzionalità di base di Botfront, come l'interfaccia dei diagrammi di flusso dei moduli e il multilinguismo,

Nel seguente verranno illustrate le seguenti fasi:

* Installazione, Configurazione e Addestramento.

## 3.1 Installazione

1. Installazione node 16
2. Eseguire il comando
   1. npm install -g botfront
3. crea una nuova cartella
4. Eseguire i comandi
   1. botfront init
   2. botfront up

## 3.2 Configurazione

1. Aprire botfront app (***ipmacchina: 8888***)
2. Creare un progetto con l’identificativo bf-xxxxx
3. Copiare l’id generato e copiarlo nel file botfront\.botfront\botfront.yml
   1. ***bf\_project\_id: identificativo***

## 3.3 Addestramento

Botfront Web offre una sezione per l’addestramento.

**Caso Primo Addestramento**:

1. Accedere alla sezione *Dialogue*
2. Accedere alla sezione *Policies* e settare le Policies con il seguente snippet:

policies:

- name: MemoizationPolicy

- name: RulePolicy

1. Accedere alla sezione *NLU* e accedere alla sezione *Settings*.
2. Settare la pipeline con la seguente configurazione

pipeline:

- name: WhitespaceTokenizer

- name: CountVectorsFeaturizer

analyzer: word

- name: CountVectorsFeaturizer

analyzer: char\_wb

min\_ngram: 2

max\_ngram: 4

- name: DIETClassifier

epochs: 50

- name: EntitySynonymMapper

- name: FallbackClassifier

threshold: 0.7

ambiguity\_threshold: 0.1

1. Accedere alla sezione *Settings* e cliccare su *Import/Export*
2. Caricare i seguenti files: *nlu.yml, rules.yml*, *domain.yml*
3. A valle del caricamento ritornare sulla sezione *NLU* e cliccare su Train
4. Se l’icona in alto dell’applicazione è una spunta verde, allora l’addestramento è andato a buon fine.

**Caso Successivo Addestramento**:

1. Accedere alla sezione *Dialogue*
2. Accedere alla sezione *Policies* e settare le Policies con il seguente snippet:

policies:

- name: MemoizationPolicy

- name: RulePolicy

1. Accedere alla sezione *NLU* e accedere alla sezione *Settings*.
2. Settare la pipeline con la seguente configurazione

pipeline:

- name: WhitespaceTokenizer

- name: CountVectorsFeaturizer

analyzer: word

- name: CountVectorsFeaturizer

analyzer: char\_wb

min\_ngram: 2

max\_ngram: 4

- name: DIETClassifier

epochs: 50

- name: EntitySynonymMapper

- name: FallbackClassifier

threshold: 0.7

ambiguity\_threshold: 0.1

1. Accedere alla sezione *Settings* e cliccare su *Import/Export*
2. Selezionare “*reset project*”
3. Caricare i seguenti files: *nlu.yml, rules.yml*, *domain.yml*
4. A valle del caricamento ritornare sulla sezione *NLU* e cliccare su Train
5. Se l’icona in alto dell’applicazione è una spunta verde, allora l’addestramento è andato a buon fine.

# Iside

Iside è l’applicativo utilizzato per la comunicazione tramite Web Socket con il portale Istituzionale e la Mobile App.

L’applicativo utilizza due Web Socket:

1. Iside – Client
2. Iside – Botfront Rasa

Il secondo Web Socket serve ad inviare le richieste del client, la risposta verrà ribaltata al Client. Nel caso in cui nella risposta ci sono delle variabili particolari, quest’ultime vengono sostituite con i valori presenti sul portale.

## 4.1 Installazione

1. Copiare la cartella *microbot* sotto la macchina
2. Navigare nelle cartelle al fine di accedere nella folder *docker*
3. Eseguire i comandi
   1. docker-compose build
   2. docker-compose up -d

## 4.2 Database Iside

Il database contiene le configurazioni necessarie ad Iside e in particolare l’endpoint per l’interfacciamento con il portale.

Accedere al server “*10.10.5.112*”, copiare nella cartella principale il file “*db\_iside.sql*” presente nella cartella “*database\_iside*”.

Dopo di che eseguire seguire i seguenti step:

1. Eseguire il comando
   1. psql -U postgres
2. inserire la password di postres
3. Eseguire i comandi
   1. create database iside;
   2. create user iside with encrypted password 'password';
   3. grant all privileges on database iside to iside;
4. Eseguire il comando
   1. psql -U iside < db\_iside.sql
   2. password

## 4.3 Configurazione

Nel file di properties presente in **“microbot\Iside\src\main\resources**” sostituire le linee 24,25,26,27 con:

spring.datasource.url=jdbc:postgresql://10.10.5.112:5432/iside

spring.datasource.username=iside

spring.datasource.password=**password**

spring.datasource.driverClassName=org.postgresql.Driver

**Configurazione Recupero Variabili dal Portale**

A valle della connessione al database da linea di comando eseguire la seguente CURL

curl --location --request POST 'http://localhost:7052/iside/admin/addsocketconfiguration' \

\

--header 'Content-Type: application/json' \

--data-raw '{

{

"type":"local",

"socketChannel":"http://10.10.5.114:5005",

"receiveEvent":"bot\_uttered",

"sendEvent":"user\_uttered",

"socketParameters":{

"type":"local",

"substitutionendpoint":"http://10.10.5.110/json/chatbot",

"method":"GET"

},

"requestBody":{

"message":"${message}",

"session\_id":"${session\_id}",

"customData":{

"language":"it"

}

}

}

'

Info CURL

* **socketChannel**: endpoint di rasa
* **substitutionendpoint**: endpoint del portale
  + Es."http://ip\_portale/json/chatbot",

Inoltre, è necessario aggiungere il file “*substitutionconfig\_socket\_local.yml*” presente in “*files\_iside*” nel container di iside.

Per effettuare tale operazione basta eseguire il seguente comando:

* docker cp iside:/app/substitutionconfig\_socket\_local.yml **folder\_dove\_risiede\_il\_file**

# nginx

nginx è utilizzato per il reverse proxy sulle componenti Iside e Botfront.

## 5.1 Installazione

Copiare la cartella nginx sotto il server e posizionarsi all’interno.

Eseguire i seguenti comandi:

1. docker build -t chatbot-nginx .
2. docker run -d --name nginx -p 80:80 chatbot-nginx

# Convertitore

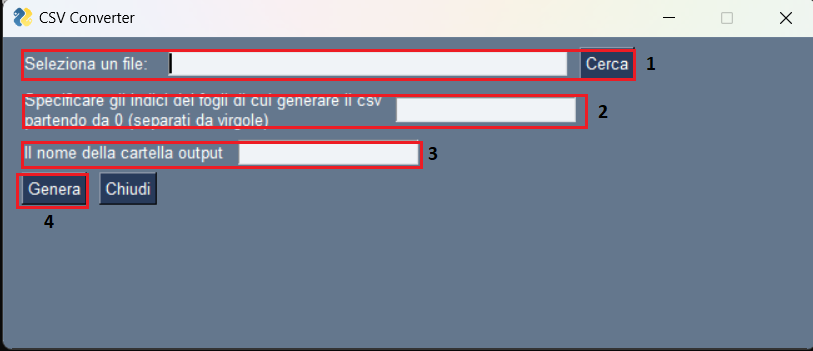
Il convertitore permette di convertire un addestramento sottoforma di Excel in tre files necessari per l’addestramento reale su Botfront.

## 6.1 Installazione

Nella cartella “*csv-converter-2.0*” ed eseguire il comando “pip install -r requirements.txt” per l’installazione delle librerie necessarie al convertitore.

## 6.2 Esecuzione

Eseguire il comando “python converter\_interface.py” e viene mostrata un’interfaccia come la seguente:



1. Selezione del file excel
2. Lasciarlo vuoto
3. A partire dalla directory corrente, specificare il nome della cartella di output
4. Genera i files

Dopo il click del pulsante “Genera”, l’applicativo genera tre files (nlu.yml, rules.yml, domain.yml) nella cartella di output specificata.

## 6.3 Istruzioni compilare Excel

1. Definire un primo foglio chiamato “Sinonimi”. La cella A1 conterrà la stringa “SINONIMI”. Le righe successive della colonna A conterranno delle liste di sinonimi separati da virgole.
2. Definire un secondo foglio chiamato “fallback”. La cella A1 conterrà la stringa “Testo di fallback”. La cella A2 conterrà la stringa che verrà mostrata in caso di fallback.
3. Definire gli intenti in fogli successivi secondo il seguente formato:

**L’intestazione del foglio sarà**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TASSONOMIA | INTENTO | DOMANDE | RISPOSTA | RISPOSTE BOTTONI | RISPOSTA IMMAGINE |

* Il campo TASSONOMIA è opzionale
* Il campo INTENTO è obbligatorio
* Scrivere le domande una sotto l’altra nella colonna DOMANDE lasciando la prima riga (quella corrispondente all’INTENTO) vuota
* Il campo RISPOSTA può contenere del testo con placeholder secondo il formato “#$placeholder$#”
* Il campo RISPOSTE BOTTONI può contenere una serie di bottoni secondo il formato “titolo”: “url” oppure “titolo”:”nome\_intento”, separati da virgole
* Il campo RISPOSTA IMMAGINE può contenere l’url ad una immagine

In caso di presenza di bottoni ed immagine, il campo RISPOSTA dovrà rispettare questo formato “titolo immagine | sottotitolo immagine”.

## 6.4 Esempio Excel

Il file “*esempio\_compilazione.xlsx*” mostra un esempio di come compilare l’excel.

# Comandi docker

Il seguente paragrafo ha l’obiettivo di illustrare i comandi necessari per la manutenzione e monitoraggio di tutto il sistema.

## 7.1 Riavvio containers

I seguenti i comandi sono utili per riavviare i vari container (applicativi):

* Botfront-app
  + docker restart botfront-app
* Botfront-rasa
  + docker restart botfront-rasa
* Botfront-mongo
  + docker restart botfront-mongo
* Iside
  + docker restart iside

## 7.2 Logs

I seguenti i comandi sono utili per visionare il log in real time dei vari container (applicativi):

* Botfront-app
  + docker logs botfront-app --tail 500 -f
* Botfront-rasa
  + docker logs botfront-rasa --tail 500 -f
* Botfront-mongo
  + docker logs botfront-mongo --tail 500 -f
* Iside
  + docker logs iside --tail 500 -f

## 7.2 Rimozione Logs

I seguenti i comandi sono utili per rimuovere i file di log di un qualsiasi container installato sulla macchina:

* Eseguire i seguenti comandi
  + sudo su
  + docker ps
    - recuperare l’id del container di interesse (prima colonna)
  + cd /var/lib/docker/containers
  + ls
    - trovare il container le cui 12 cifre sono uguale all’id del container di interesse
  + cd xxxxxxxxxxxxyyyyyyyyy…y
  + rm xxxxxxxxxxxxyyyyyyyyy…y -json.log
  + docker restart **idcontainer**