



ONION

S O F T W A R E

softwareonion@gmail.com

Manuale utente

Informazioni sul documento

Versione	v2.0.0
Data approvazione	2019-07-12
Responsabili	Matteo Lotto
Redattori	Nicola Zorzo Matteo Lotto Nicola Pastore
Verificatori	Federico Omodei
Stato	approvato
Lista distribuzione	<i>Onion Software</i> <i>Imola Informatica S.p.A.</i> <i>prof. Tullio Vardanega</i> <i>prof. Riccardo Cardin</i>

Scopo del documento

Manuale utente del progetto Butterfly.

Registro delle modifiche

Versione	Data	Autore	Ruolo	Descrizione
v2.0.0	2019-07-12	Matteo Lotto	Responsabile	Documento approvato per il rilascio
v1.1.0	2019-07-03	Federico Omodei	Verificatore	Verifica superata
v1.0.2	2019-06-27	Nicola Zorzo	Progettista	Rettifica documento secondo indicazioni
v1.0.1	2019-06-24	Nicola Zorzo	Progettista	Integrazione immagini esplicative modificate
v1.0.0	2019-06-08	Linpeng Zhang	Responsabile	Documento approvato per il rilascio provvisorio
v0.1.0	2019-06-05	Nicola Pastore	Verificatore	Verifica superata
v0.0.5	2019-06-03	Nicola Zorzo	Progettista	Integrazione immagini esplicative
v0.0.4	2019-05-31	Nicola Zorzo	Progettista	Stesura §4 §5
v0.0.3	2019-05-28	Federico Omodei	Progettista	Stesura §3
v0.0.2	2019-05-27	Matteo Lotto	Progettista	Stesura §1 §2
v0.0.1	2019-05-24	Nicola Zorzo	Progettista	Creazione del documento

Indice

1	Introduzione	1
1.1	Glossario	1
1.2	Scopo del prodotto	1
2	Configurazione	2
2.1	Requisiti di sistema	2
2.1.1	Software	2
2.1.2	Hardware	2
2.2	Suddivisione container	3
2.3	Mappatura porte	3
2.4	Avvio di Butterfly	4
2.5	Configurazione webhook	4
2.5.1	Redmine	4
2.5.2	GitLab	7
2.6	Configurazione Butterfly	10
2.6.1	Producer	10
2.6.1.1	redmine-producer	11
2.6.1.2	gitlab-producer	11
2.6.2	Consumer	11
2.6.2.1	telegram-consumer	12
2.6.2.2	email-consumer	12
2.6.2.3	slack-consumer	13
2.6.3	Gestore personale	13
2.6.4	API ed interfaccia web	14
3	Utilizzo di Butterfly	15
3.1	Gestore personale	15
3.1.1	Note informative	15
3.1.2	Interfaccia web	15
3.1.2.1	Home page	15
3.1.2.2	Contatti	16
3.1.2.2.1	Visualizzazione elementi	16
3.1.2.2.2	Inserimento	17
3.1.2.2.3	Modifica	19
3.1.2.2.4	Eliminazione	20
3.1.2.2.5	Ricerca	21
3.1.2.2.6	Visualizzazione errori	21
3.1.2.3	Progetti	22
3.1.2.3.1	Visualizzazione elementi	22
3.1.2.3.2	Inserimento	23
3.1.2.3.3	Modifica	25
3.1.2.3.4	Eliminazione	26
3.1.2.3.5	Ricerca	27
3.1.2.3.6	Visualizzazione errori	27
3.1.2.4	Iscrizioni	28
3.1.2.4.1	Visualizzazione elementi	28
3.1.2.4.2	Inserimento	29
3.1.2.4.3	Modifica	31
3.1.2.4.4	Eliminazione	32
3.1.2.4.5	Ricerca	33
3.1.2.4.6	Visualizzazione errori	33
3.2	RESTful API	34

3.2.1	Contatti	34
3.2.1.1	Lista	34
3.2.1.2	Inserimento	35
3.2.1.3	Aggiornamento	36
3.2.1.4	Cancellazione	37
3.2.2	Progetti	38
3.2.2.1	Lista	38
3.2.2.2	Inserimento	38
3.2.2.3	Aggiornamento	39
3.2.2.4	Cancellazione	40
3.2.3	Iscrizioni	40
3.2.3.1	Lista	40
3.2.3.2	Inserimento	41
3.2.3.3	Aggiornamento	42
3.2.3.4	Cancellazione	43
3.3	Ricezione notifiche	43
3.4	Nota sulla ricezione delle notifiche	45
4	Contatti	46
4.1	Segnalazioni e suggerimenti	46
4.1.1	Github	46
4.1.2	Email	47
4.2	Contribuire allo sviluppo	47
A	Glossario	48

Elenco delle figure

1	Reset password di Redmine	5
2	Homepage di Redmine	5
3	Pannello d'amministrazione	6
4	Interfaccia Custom Workflows	6
5	Workflow di Issue Webhook	7
6	<i>Admin Area</i> di GitLab	8
7	Impostazioni <i>Network</i>	8
8	Abilitazione <i>webhook</i>	9
9	Lista progetti di GitLab	9
10	<i>Sidebar</i> di GitLab	9
11	<i>Settings</i> di GitLab	9
12	Creazione <i>webhook</i> su GitLab	10
13	Conferma creazione <i>webhook</i> su GitLab	10
14	<i>Webhook</i> su GitLab	10
15	Home page di <i>Butterfly</i>	15
16	Sezione CONTACTS	16
17	Sezione CONTACTS (vuota)	17
18	Pulsante <i>INSERT CONTACT</i>	17
19	<i>Modal</i> di inserimento contatto	18
20	Conferma di inserimento contatto	18
21	Visualizzazione contatto inserito	19
22	Pulsante di modifica contatto	19
23	<i>Modal</i> di modifica contatto	20
24	Pulsante di eliminazione contatto	20
25	<i>Modal</i> di eliminazione contatto	21
26	Box di ricerca contatti	21
27	Visualizzazione contatti ricercati	21

28	Visualizzazione messaggio d'errore	22
29	Sezione PROJECTS	22
30	Sezione PROJECTS (vuota)	23
31	Pulsante INSERT PROJECT	23
32	<i>Modal</i> di inserimento progetto	24
33	Conferma di inserimento progetto	24
34	Visualizzazione progetto inserito	25
35	Pulsante di modifica progetto	25
36	<i>Modal</i> di modifica progetto	26
37	Pulsante di eliminazione progetto	26
38	<i>Modal</i> di eliminazione progetto	27
39	Box di ricerca progetti	27
40	Visualizzazione progetti ricercati	27
41	Visualizzazione messaggio d'errore	28
42	Sezione SUBSCRIPTIONS	28
43	Sezione SUBSCRIPTIONS (vuota)	29
44	Pulsante INSERT SUBSCRIPTION	29
45	<i>Modal</i> di inserimento iscrizione	30
46	Conferma di inserimento iscrizione	30
47	Visualizzazione iscrizione inserita	31
48	Pulsante di modifica iscrizione	31
49	<i>Modal</i> di modifica iscrizione	32
50	Pulsante di eliminazione iscrizione	32
51	<i>Modal</i> di eliminazione iscrizione	33
52	Box di ricerca iscrizioni	33
53	Visualizzazione iscrizioni ricercate	33
54	Visualizzazione messaggio d'errore	34
55	Esempio di notifica email	44
56	Esempio di notifica Telegram	44
57	Esempio di notifica Slack	44
58	Inserimento di una issue su Github	46

Elenco delle tabelle

2	Requisiti software	2
3	Immagini software fornite	2
4	Requisiti hardware	2
5	Mappatura porte per <i>container</i>	4



1 Introduzione

1.1 Glossario

Al fine di evitare possibili ambiguità relative al linguaggio utilizzato nel documento, viene fornito un glossario in appendice §A, in cui vengono definiti e descritti tutti i termini con un significato specifico. Per facilitare la comprensione, i termini presenti in questo glossario saranno contrassegnati da una “G” a pedice.

1.2 Scopo del prodotto

Butterfly è un applicativo che permette la gestione dinamica ed intelligente di notifiche provenienti da strumenti utilizzati in ambito *DevOps*. Il software permette di incapsulare le segnalazioni provenienti da GitLab e Redmine, filtrarle tramite il gestore personale di *Butterfly*, ed inoltrarle, a seconda della priorità, disponibilità ed interesse, alla persona più adeguata tramite email, Telegram o Slack (in base alla preferenza espressa nel gestore personale).



2 Configurazione

Per poter utilizzare *Butterfly* è necessaria una configurazione preliminare. Su richiesta della proponente *Imola Informatica S.p.A.*, tutti i servizi del prodotto sono stati suddivisi in *Docker container* per facilitarne l'utilizzo e la divisione logica. Inoltre, per agevolare l'avvio contemporaneo dei servizi, viene utilizzato *docker-compose*.

2.1 Requisiti di sistema

Di seguito sono riportati i requisiti **minimi** necessari per un corretto funzionamento del prodotto *Butterfly*. Questi requisiti tengono conto della natura del prodotto e della forte integrazione di quest'ultimo con sistemi di terze parti.

2.1.1 Software

Software	Versione utilizzata
Docker	18.06
docker-compose	1.24

Tabella 2: Requisiti software

Per tutti gli altri componenti software che fanno parte integrante del nostro prodotto vengono fornite le rispettive immagini di Docker, di conseguenza non è necessario soddisfare altri requisiti oltre a quelli precedentemente riportati.

Immagine	Versione utilizzata
ZooKeeper	3.4
Kafka	2.2.1
Redmine	3.4.6
GitLab	12.0
MongoDB	4.0.10

Tabella 3: Immagini software fornite

2.1.2 Hardware

Hardware	Caratteristiche minime
OS	Ubuntu 16.04, macOS Sierra 10.12
CPU	Intel Core i5, 2.00 GHz
RAM	8.00 GB
Spazio su disco	5.00 GB
Rete	Possibilità di comunicazione tra un determinato set di porte

Tabella 4: Requisiti hardware



Ribadiamo che i requisiti hardware tengono in considerazione i requisiti minimi richiesti dai software precedentemente elencati.

2.2 Suddivisione container

Butterfly è un software che può essere suddiviso in base ai servizi che offre, ognuno dei quali viene rappresentato da un Docker *container*.

La suddivisione logica dei *container* utilizzati è la seguente:

- **zookeeper**: avvia il server di riferimento ZooKeeper su cui fa appoggio Kafka (immagine di base: `zookeeper`);
- **kafka**: avvia il *broker* Kafka (immagine di base: `wurstmeister/kafka`);
- **redmine-producer**: avvia il *producer* di Redmine (immagine custom: `redmine-producer`);
- **gitlab-producer**: avvia il *producer* di GitLab (immagine custom: `gitlab-producer`);
- **email-consumer**: avvia il *consumer* per le email (immagine custom: `email-consumer`);
- **telegram-consumer**: avvia il *consumer* per Telegram (immagine custom: `telegram-consumer`);
- **slack-consumer**: avvia il *consumer* per Slack (immagine custom: `slack-consumer`);
- **manager**: avvia il gestore personale (immagine custom: `manager`);
- **web-app**: avvia l’interfaccia web (immagine custom: `web-app`);
- **mongo**: avvia il database MongoDB su cui vengono immagazzinati i dati (immagine di base: `mongo`);
- **gitlab**: avvia il servizio GitLab (immagine di base: `gitlab/gitlab-ce`);
- **redmine**: avvia il servizio Redmine (immagine di base: `redmine`).

Ogni immagine di base utilizzata è reperibile su [Docker Hub](#). Per la costruzione dei componenti custom che fanno parte di *Butterfly* sono stati creati dei *Dockerfile_G* specifici, e di conseguenza delle nuove immagini (individuate dal termine “immagine custom”). Quest’ultime sono disponibili solamente all’interno del [repository](#) del gruppo *Onion Software*.

2.3 Mappatura porte

Per evitare conflitti con altri servizi, è stato assegnato ad ogni *container* un numero di porte pari al numero di porte che espone (i *container* che non espongono porte sono stati omessi).



Container	Porte esposte	Porte assegnate
zookeeper	2181	2181
kafka	9092	9092
redmine-producer	5000	30020
gitlab-producer	5000	30021
web-app	80	30000
mongo	27017	27017
gitlab	80	4080
redmine	3000	8080

Tabella 5: Mappatura porte per *container*

2.4 Avvio di Butterfly

Dopo essersi assicurati di rispettare i requisiti hardware e software è necessario avviare i servizi (l'operazione d'istanziazione dei *container* può richiedere qualche minuto):

1. *Butterfly*: da terminale, dirigersi nella cartella `Butterfly/Butterfly/butterflyApp` ed eseguire il comando `docker-compose up -d`;
2. Redmine: da terminale, dirigersi nella cartella `Butterfly/Butterfly/redmineServer` ed eseguire il comando `docker-compose up -d`;
3. GitLab: da terminale, dirigersi nella cartella `Butterfly/Butterfly/gitlabServer` ed eseguire il comando `docker-compose up -d`.

Il flag `-d` fa partire i container in *detached mode* per non far visualizzare log inutili durante la loro esecuzione, di conseguenza la sua presenza è opzionale.

I servizi di Redmine e GitLab vengono forniti per poter testare il prodotto simulando l'ambiente di sviluppo della proponente *Imola Informatica S.p.A.* e sono completamente configurabili.

2.5 Configurazione webhook

Per un funzionamento corretto dei *webhook*, ogni servizio di terze parti che comunica con i nostri *producer* ha bisogno di una configurazione preliminare.

2.5.1 Redmine

Di seguito sono elencati i passaggi necessari per configurare l'invio dei *webhook* da parte di Redmine attraverso l'utilizzo di un *plugin* esterno chiamato *Custom Workflows*, che è necessario installare nel *container* di Redmine attraverso alcuni semplici passaggi:

1. aprire un terminale, verificare che il servizio di Redmine sia attivo attraverso il comando `docker ps` e cercare la presenza di un *container* denominato `redmine`;
2. eseguire il comando `docker exec -t -i redmine bash` per accedere al *container*;
3. eseguire il comando `ls` e verificare la presenza di una cartella denominata `plugins`, dopodichè spostarsi su quella cartella eseguendo `cd plugins`;
4. eseguire il comando `git clone https://github.com/Onion-Software/redmine_custom_workflows.git` e poi verificare la presenza di una cartella denominata `redmine_custom_workflows` tramite il comando `ls`;



5. tornare nel *path* iniziale `/usr/src/redmine` ed eseguire il comando `rake redmine:plugins:migrate;`
6. il *plugin* è stato installato, uscire dal *container* eseguendo `exit`;
7. da terminale, dirigersi nella cartella `Butterfly/Butterfly/redmineServer` ed eseguire il comando `docker-compose restart` per riavviare il servizio di Redmine con il plugin installato.

Dopo l'installazione del *plugin* si può utilizzare il servizio di Redmine con tutte le sue funzionalità:

1. eseguire l'accesso a Redmine all'indirizzo `localhost:8080` e autenticarsi con un account di tipo amministratore, utilizzando le seguenti credenziali:
 - username: `admin`;
 - password: `admin`.

Dopo il primo *login* apparirà un messaggio che richiederà di resettare la password; inserirne una a piacimento;

Your password has expired or the administrator requires you to change it.

Change password

Password * New password *
Must be at least 8 characters long.
Confirmation *

Apply

Figura 1: Reset password di Redmine

2. aprire il pannello di amministrazione con il link “Administration” in alto a sinistra;

Home My page Projects Administration Help

Redmine

Home

Powered by Bitnami Redmine Stack © 2006–2018 Jean-Philippe Lang

Figura 2: Homepage di Redmine



3. selezionare l'opzione “Custom workflows” nell'elenco che compare;

The screenshot shows the Redmine administration interface. At the top, there's a navigation bar with links for Home, My page, Projects, Administration, and Help. On the right side of the header, it says "Logged in as [username] My account Sign out". Below the header is a search bar labeled "Search:" and a dropdown menu labeled "Jump to a project...". The main content area has a sidebar titled "Administration" containing links for Projects, Users, Groups, Roles and permissions, Trackers, Issue statuses, Workflow, Custom fields, Enumerations, Settings, LDAP authentication, Plugins, and Information. The "Custom workflows" link is highlighted with a green background. At the bottom of the page, there's a footer bar with the text "Powered by Bitnami Redmine Stack © 2006-2018 Jean-Philippe Lang".

Figura 3: Pannello d'amministrazione

4. una volta aperto l'interfaccia del plugin, selezionare “Import workflow” e scegliere il file “Issue Webhook.xml” presente nella cartella **Butterfly/Butterfly/redmineServer**;

The screenshot shows the Redmine Custom Workflows interface. At the top, there's a navigation bar with links for Home, My page, Projects, Administration, and Help. On the right side of the header, it says "Logged in as supervisor My account Sign out". Below the header is a search bar labeled "Search:" and a dropdown menu labeled "Jump to a project...". The main content area has a table titled "Custom workflows" with columns: Name, Description, Observable object, Author, Projects, and Sort. There are two rows in the table: 1) "Duration/Done Ratio/Status correlation" which describes setting up a correlation between start date, due date, done ratio, and status of issues, and lists several conditions for status changes. 2) "Issue Webhook" which describes sending a POST to a specific URL for issue modifications. To the right of the table is an "Administration" sidebar with links for Projects, Users, Groups, Roles and permissions, Trackers, Issue statuses, Workflow, Custom fields, Enumerations, Settings, LDAP authentication, Plugins, and Information. The "Plugins" section is highlighted with a green background. At the bottom of the page, there's a footer bar with the text "Powered by Bitnami Redmine Stack © 2006-2018 Jean-Philippe Lang".

Figura 4: Interfaccia Custom Workflows

5. selezionare per quali progetti si vuole utilizzare il *plugin* o, in alternativa, selezionare l'opzione “Enabled for all projects”;



The screenshot shows the Redmine 'Custom workflows' interface. On the left, there's a sidebar with links like 'Projects', 'Activity', 'Issues', 'Spent time', 'Gantt', 'Calendar', 'News', 'Administration' (with sub-links for 'Projects', 'Users', 'Groups', 'Roles and permissions', 'Trackers', 'Issue statuses', 'Workflow', 'Custom fields', 'Enumerations', 'Settings', 'LDAP authentication', 'Plugins', and 'Information'), and 'Information'. The main area is titled 'Custom workflows > Issue Webhook'. It has fields for 'Name' (set to 'Issue Webhook'), 'Author's e-mail', 'Enabled for project(s)' (with 'TestProject' selected), 'Check all | Uncheck all', 'Observable object' (set to 'Issue'), 'Description' (containing text about sending a POST request to an issue modification URL), and checkboxes for 'Enabled for all projects' and 'Active'. Below this is a section for 'Workflow scripts' with two code snippets: 'Workflow script executable before saving observable object' and 'Workflow script executable after saving observable object'. The 'Workflow script executable after saving observable object' contains the following JavaScript:

```
require 'net/http'
require 'uri'
require 'json'

url = URI.parse("http://webhook.site/13f460a5-f757-4cde-a891-1ef1b83881f8")

header = {'Content-Type': 'text/json'}
user = {
  "object_kind": "issue",
  "event_type": "issue",
  "subject": subject,
  "description": description,
  "status_id": status,
  "created_on": created_on,
  "updated_on": updated_on}
```

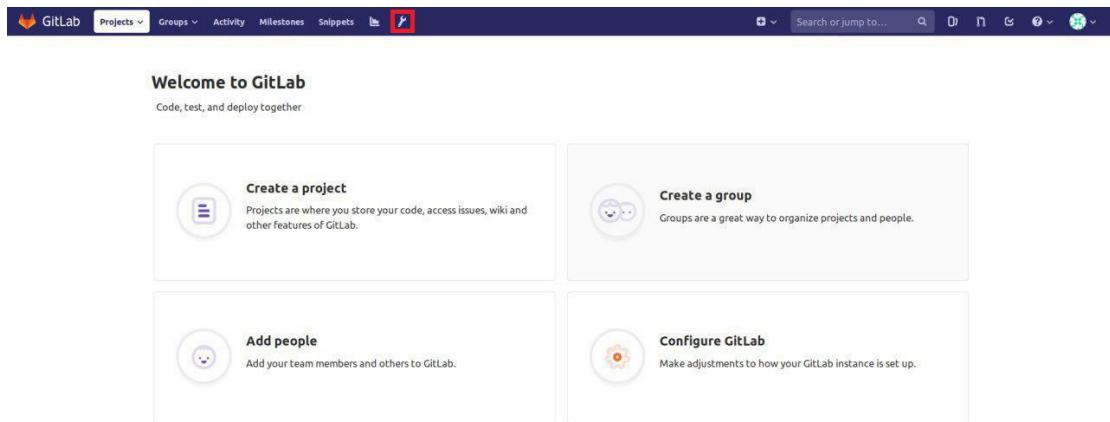
Figura 5: Workflow di Issue Webhook

6. una volta selezionati i progetti, salvato il workflow e tornati nell'interfaccia di Custom workflows, selezionare l'opzione “**Activate**” a destra dell'*Issue Webhook*.

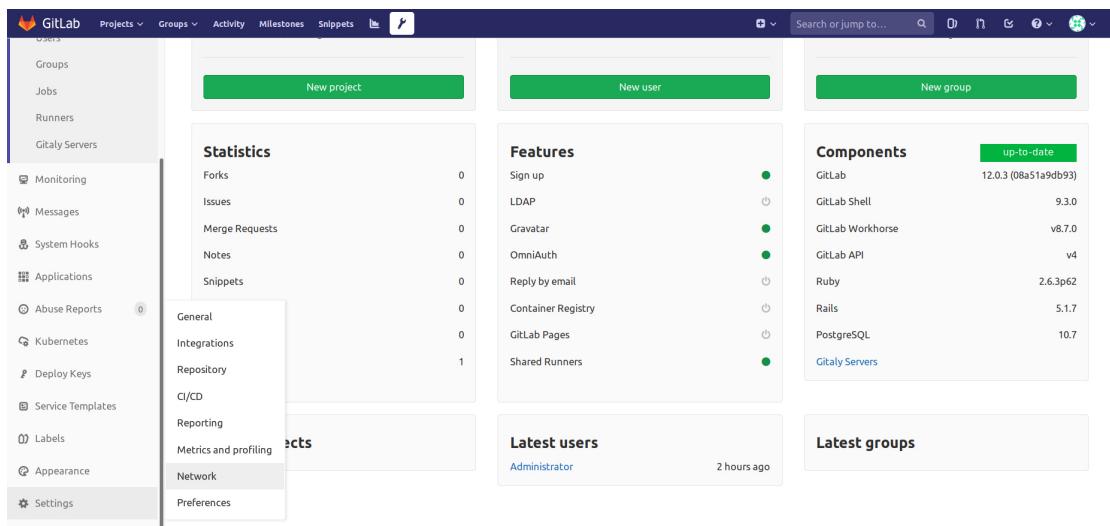
2.5.2 GitLab

L'immagine di GitLab è stata fornita perché il sito ufficiale non supporta l'invio di *webhook* a macchine che risiedono all'interno della stessa rete, di conseguenza non sarebbe stato possibile testarne la funzionalità. Di seguito, quindi, sono elencati i passaggi preliminari (da 1 a 4, per portare a termine l'operazione è **necessario accedere con un account amministratore**) e quelli necessari (da 5 in poi) per configurare l'invio dei *webhook* da parte del servizio di GitLab:

1. eseguire l'accesso a GitLab all'indirizzo `localhost:4080` e autenticarsi con un account di tipo amministratore. La prima volta che la pagina viene visitata viene richiesto di cambiare la password per l'account amministratore del sistema, dopodichè è possibile effettuare il login utilizzando le seguenti credenziali:
 - username: `root`;
 - password: quella appena impostata;
2. una volta loggati, accedere all'*Admin Area* in alto;

Figura 6: *Admin Area* di GitLab

3. dalla *sidebar* selezionare l'opzione *Settings* e poi *Network*;

Figura 7: Impostazioni *Network*

4. nella voce *Outbound requests* spuntare *Allow requests to the local network from hooks and services* e salvare;



The screenshot shows the 'Admin Area > Network' section of the GitLab interface. The 'Outbound requests' section is expanded, displaying two configuration options:

- Allow requests to the local network from hooks and services
- Enforce DNS rebinding attack protection

A green 'Save changes' button is located at the bottom right.

Figura 8: Abilitazione *webhook*

5. selezionare il progetto per cui si vogliono abilitare i *webhook* tra quelli disponibili nella sezione *Projects*;

The screenshot shows the 'Projects' page with a list of projects. The project 'Nicola Zorzo/test-project' is highlighted with a red box. Other visible projects are 'Onion Software/Butterfly' and 'Nicola Zorzo/test-project'.

Figura 9: Lista progetti di GitLab

6. selezionare nel menu a sinistra l'opzione *Settings* e successivamente l'opzione *Integrations*;

The screenshot shows the left sidebar of the GitLab interface. The 'Settings' option is highlighted with a red box. The 'Integrations' option is also visible in the list.

Figura 10: Sidebar di GitLab

Figura 11: *Settings* di GitLab

7. creare il *webhook* secondo le proprie necessità e in base agli eventi a cui si è interessati e confermarne la creazione. Per tutti i webhook è necessario inserire come URL l'indirizzo `http://gitlab-producer:5000/webhook`;



The screenshot shows the 'Integrations Settings' page for a project named 'test'. On the left, there is a sidebar with 'Settings' selected under 'Integrations'. The main area displays the 'Integrations' section with a brief description of what webhooks are used for. A 'URL' field contains 'http://gitlab-producer:5000/webhook'. A 'Secret Token' field is present but empty. The 'Trigger' section contains several checkboxes: 'Push events' (checked), 'Tag push events', 'Comments', 'Confidential Comments', 'Issues events', 'Confidential Issues events', and 'Merge request events'. Below these is a note about sending the X-GitLab-Token header. A large green 'Add webhook' button is at the bottom.

Figura 12: Creazione *webhook* su GitLab

This screenshot shows the confirmation step for adding a webhook. It lists two selected triggers: 'Pipeline events' and 'Wiki Page events'. Both have descriptive notes below them. Under 'SSL verification', 'Enable SSL verification' is checked. At the bottom is a large green 'Add webhook' button.

Figura 13: Conferma creazione *webhook* su GitLab

8. verificare la presenza del *webhook* appena creato.

The screenshot shows the 'Webhooks' page with one entry. The webhook URL is 'http://gitlab-producer:5000/webhook'. It has checkboxes for 'Push Events', 'Tag Push Events', 'Issues Events', and 'Pipeline Events', all of which are checked. The 'SSL Verification' status is 'enabled'. There are 'Edit', 'Test', and 'Delete' buttons to the right of the URL.

Figura 14: *Webhook* su GitLab

2.6 Configurazione Butterfly

I componenti di Butterfly hanno un apposito file di configurazione *config.json* che può essere modificato per personalizzare alcune impostazioni. Di seguito riportiamo le istruzioni relative.

2.6.1 Producer

Tutti i componenti producer devono inviare opportuni messaggi a Kafka. Sono configurabili:

- nome dei topic su cui inviare i messaggi modificando il valore relativo alla chiave *topic_producer_name_list*;
- ip e porta del broker modificando il valore relativo alla chiave *bootstrap_servers*;

Inoltre ogni specifico producer è ulteriormente personalizzabile. In particolare:

- il nome che si vuole assegnare al server in ascolto dei webhook è personalizzabile modificando il valore relativo alla chiave *app_name*;



- l'ip del server in ascolto dei webhook è personalizzabile modificando il valore relativo alla chiave *ip*;
- la porta del server in ascolto dei webhook è personalizzabile modificando il valore relativo alla chiave *porta*.

Si riportano di seguito alcuni esempi di configurazioni.

2.6.1.1 redmine-producer

```
{  
    "kafka_topics": {  
        "topic_producer_name_list": {  
            "redmine": "redmine"  
        }  
    },  
    "kafka_producer": {  
        "bootstrap_servers": "localhost:9092"  
    },  
    "gitlab": {  
        "app_name": "Redmine",  
        "ip": "0.0.0.0",  
        "port": "5000"  
    }  
}
```

Listing 1: Esempio di configurazione relativa a redmine-producer

2.6.1.2 gitlab-producer

```
{  
    "kafka_topics": {  
        "topic_producer_name_list": {  
            "gitlab": "gitlab"  
        }  
    },  
    "kafka_producer": {  
        "bootstrap_servers": "localhost:9092"  
    },  
    "gitlab": {  
        "app_name": "GitLab",  
        "ip": "0.0.0.0",  
        "port": "5000"  
    }  
}
```

Listing 2: Esempio di configurazione relativa a gitlab-producer

2.6.2 Consumer

Tutti i componenti consumer devono ascoltare i messaggi su opportuni topic di Kafka. Sono configurabili:

- nome dei topic su cui rimanere in ascolto modificando il valore relativo alla chiave *topic_consumer_name_list*;



- ip e porta del broker modificando il valore relativo alla chiave `bootstrap_servers`;
- la modalità con cui si incrementa l'offset relativo ai messaggi in un topic modificando il valore relativo alla chiave `enable_auto_commit`;
- l'intervallo di tempo con cui controllare l'arrivo di messaggi modificando il valore relativo alla chiave `auto_commit_interval_ms`;
- la possibilità di scegliere quando resettare l'offset del messaggio in un topic modificando il valore relativo alla chiave `auto_offset_reset`;
- `group_id` del consumer modificando il valore relativo alla chiave `my-group`.

Inoltre ogni specifico consumer è ulteriormente personalizzabile.

2.6.2.1 telegram-consumer

Questo componente si avvale di un bot per inviare i messaggi su Telegram. Il token del bot è configurabile modificando il valore relativo alla chiave `bot_token`.

```
{  
    "kafka_topics": {  
        "topic_consumer_name_list": {  
            "telegram": "telegram"  
        }  
    },  
    "kafka_consumer": {  
        "bootstrap_servers": "localhost:9092",  
        "enable_auto_commit": true,  
        "auto_commit_interval_ms": 300,  
        "auto_offset_reset": "earliest",  
        "group_id": "my-group"  
    },  
    "telegram": {  
        "bot_token": "123456789ABCDEFGHJKLMNOPQRSTUVWXYZ"  
    }  
}
```

Listing 3: Esempio di configurazione relativa a telegram-consumer

2.6.2.2 email-consumer

Questo componente si avvale di un server smtp per inviare email. Sono quindi personalizzabili:

- l'indirizzo del server smtp modificando il valore relativo alla chiave `smtp`;
- la porta del server smtp modificando il valore relativo alla chiave `port`;
- l'indirizzo email del mittente delle mail modificando il valore relativo alla chiave `sender`;
- la password del mittente delle mail modificando il valore relativo alla chiave `password`.

```
{  
    "kafka_topics": {  
        "topic_consumer_name_list": {  
            "email": "email"  
        }  
    },  
    "kafka_consumer": {  
        "bootstrap_servers": "localhost:9092",  
        "enable_auto_commit": true,  
        "auto_commit_interval_ms": 300,  
        "auto_offset_reset": "earliest",  
        "group_id": "my-group"  
    }  
}
```



```
    "bootstrap_servers": "localhost:9092",
    "enable_auto_commit": true,
    "auto_commit_interval_ms": 300,
    "auto_offset_reset": "earliest",
    "group_id": "my-group"
},
"email": {
    "smtp": "smtp.gmail.com",
    "port": 587,
    "sender": "email@gmail.com",
    "password": "PASSWORD"
}
}
```

Listing 4: Esempio di configurazione relativa a email-consumer

2.6.2.3 slack-consumer

Questo componente si avvale di un bot per inviare i messaggi su Slack. Il token del bot è configurabile modificando il valore relativo alla chiave *bot_token*.

```
{
    "kafka_topics": {
        "topic_consumer_name_list": {
            "slack": "slack"
        }
    },
    "kafka_consumer": {
        "bootstrap_servers": "localhost:9092",
        "enable_auto_commit": true,
        "auto_commit_interval_ms": 300,
        "auto_offset_reset": "earliest",
        "group_id": "my-group"
    },
    "slack": {
        "bot_token": "123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    }
}
```

Listing 5: Esempio di configurazione relativa a slack-consumer

2.6.3 Gestore personale

Il gestore personale possiede all'interno un consumer ed un producer, e deve accedere al database per individuare i destinatari interessati. Nella chiave *kafka_topics* sono configurabili:

- nome dei topic su cui inviare i messaggi modificando il valore relativo alla chiave *topic_producer_name_list*;
- nome dei topic su cui rimanere in ascolto modificando il valore relativo alla chiave *topic_consumer_name_list*.

Nella chiave *kafka_producer* sono configurabili:

- nome dei topic su cui inviare i messaggi modificando il valore relativo alla chiave *topic_producer_name_list*;
- ip e porta del broker modificando il valore relativo alla chiave *bootstrap_servers*.



Nella chiave *kafka-consumer* sono configurabili:

- nome dei topic su cui rimanere in ascolto modificando il valore relativo alla chiave *topic_consumer_name_list*;
- ip e porta del broker modificando il valore relativo alla chiave *bootstrap_servers*;
- la modalità con cui si incrementa l'offset relativo ai messaggi in un topic modificando il valore relativo alla chiave *enable_auto_commit*;
- l'intervallo di tempo con cui controllare l'arrivo di messaggi modificando il valore relativo alla chiave *auto_commit_interval_ms*;
- la possibilità di scegliere quando resettare l'offset del messaggio in un topic modificando il valore relativo alla chiave *auto_offset_reset*;
- *group_id* del consumer modificando il valore relativo alla chiave *my-group*.

Nella chiave *mongo* sono configurabili:

- nome del database modificando il valore relativo alla chiave *db_name*;
- ip del database modificando il valore relativo alla chiave *host*;
- porta del database modificando il valore relativo alla chiave *port*.

2.6.4 API ed interfaccia web

La web-app si occupa della gestione delle API REST e dell'interfaccia web. Naturalmente dovrà avere accesso al database, per cui nella chiave *mongo* sono configurabili:

- nome del database modificando il valore relativo alla chiave *db_name*;
- ip del database modificando il valore relativo alla chiave *host*;
- porta del database modificando il valore relativo alla chiave *port*.

Inoltre nella chiave *application* sono configurabili:

- nome della web-app modificando il valore relativo alla chiave *app_name*;
- ip della web-app modificando il valore relativo alla chiave *ip*;
- porta della web-app modificando il valore relativo alla chiave *port*.

```
{  
    "mongo": {  
        "db_name": "test-db",  
        "host": "localhost",  
        "port": 27017  
    },  
    "application": {  
        "app_name": "web-app",  
        "ip": "0.0.0.0",  
        "port": "80"  
    }  
}
```

Listing 6: Esempio di configurazione relativa a web-app



3 Utilizzo di *Butterfly*

3.1 Gestore personale

Il gestore personale è il componente principale di *Butterfly*, in quanto permette l'immagazzinamento di tutti i dati relativi a contatti, progetti e iscrizioni e filtra tutte le notifiche provenienti dai *webhook* per poi permettere a Kafka di inviarle ai destinatari adeguati. Il gruppo *Onion Software* ha deciso di sviluppare questo componente come un'interfaccia web per rendere più intuitiva possibile l'interazione dell'utente con tutte le sue funzionalità.

3.1.1 Note informative

Il gestore personale non prevede una divisione tra utenti amministratori e non, come già discusso e approvato con la proponente *Imola Informatica S.p.A.*. Questa decisione deriva dal fatto che l'utilizzo del software *dovrebbe* essere limitato alle persone che fanno parte dell'azienda e le informazioni contenute al suo interno non rappresentano dati sensibili.

3.1.2 Interfaccia web

3.1.2.1 Home page

L'home page del sito si presenta come segue. Non è richiesto alcun tipo di login.

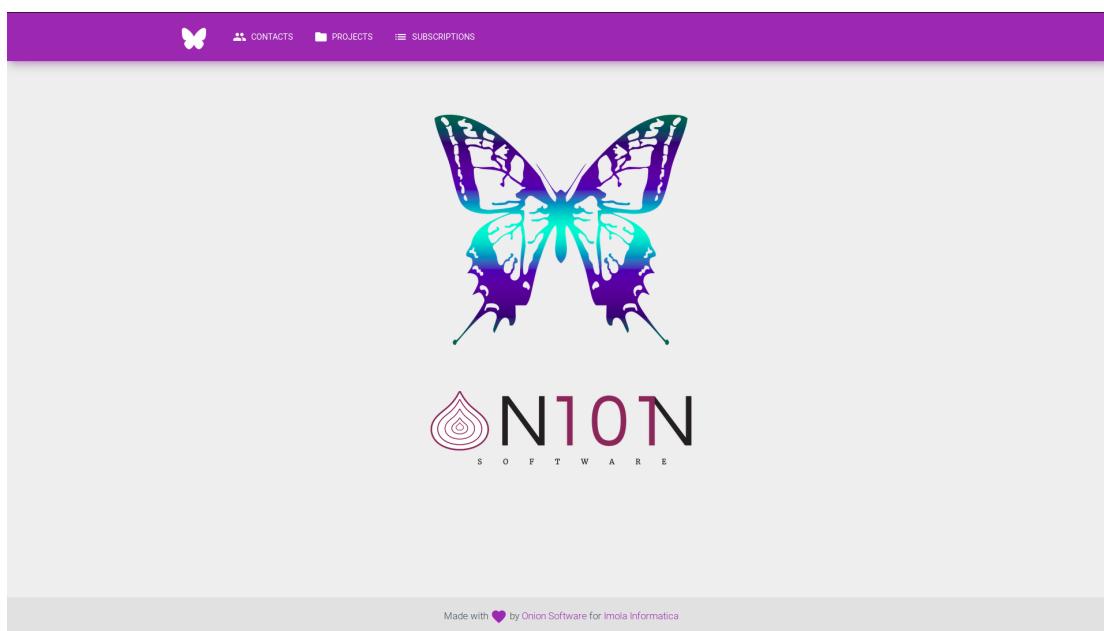


Figura 15: Home page di *Butterfly*

Nella *navbar* si possono notare le tre sezioni principali che compongono il sito:

- CONTACTS;
- PROJECTS;
- SUBSCRIPTIONS.

Ogni sezione presenta le funzionalità di:

- visualizzazione elementi;
- inserimento;



- modifica;
- eliminazione;
- ricerca;
- visualizzazione errori.

3.1.2.2 Contatti

3.1.2.2.1 Visualizzazione elementi

Questa sezione contiene la lista di tutti i contatti presenti all'interno del gestore personale.

The screenshot shows the 'CONTACTS' tab selected in the top navigation bar. Below the header, there is a search bar and a purple 'INSERT CONTACT' button. The main area displays a table with three rows of contact information:

Name	Surname	Email	Action
Mario	Rossi	mario.rossi@gmail.com	edit, delete
Luigi	Bianchi	luigi.bianchi@live.it	edit, delete
Marco	Verdi	marco.verdi@yahoo.com	edit, delete

At the bottom of the screen, a footer bar contains the text 'Made with ❤ by Onion Software for Imola Informatica'.

Figura 16: Sezione CONTACTS

Nel caso in cui non sia presente alcun contatto, verrà visualizzato un apposito messaggio.

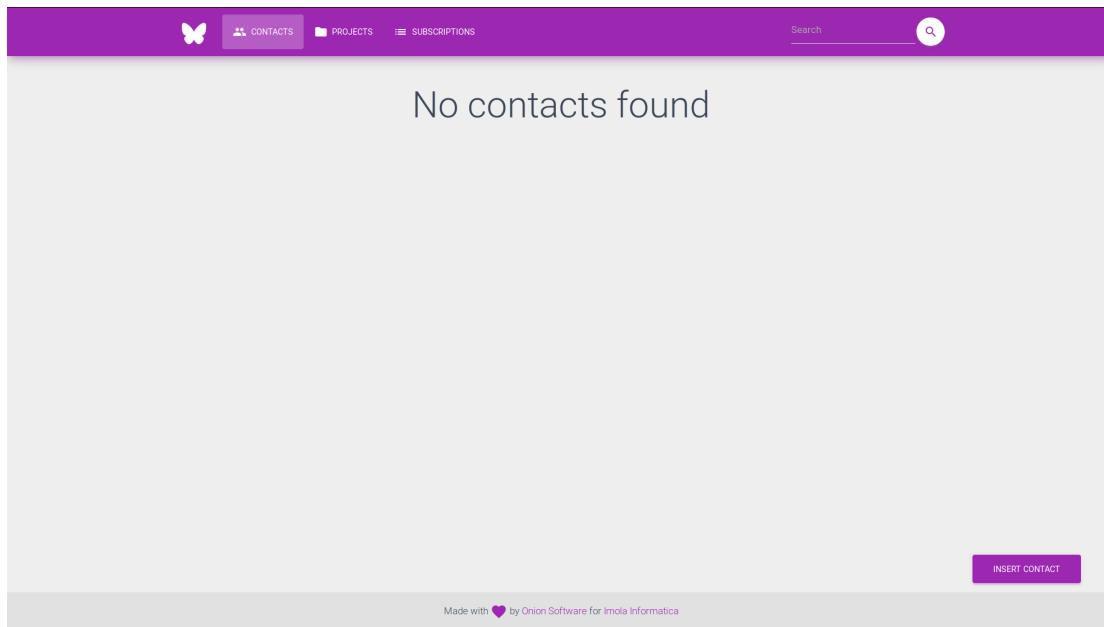


Figura 17: Sezione CONTACTS (vuota)

3.1.2.2.2 Inserimento

Per inserire un nuovo contatto:

1. premere il pulsante INSERT CONTACT situato in basso a destra;

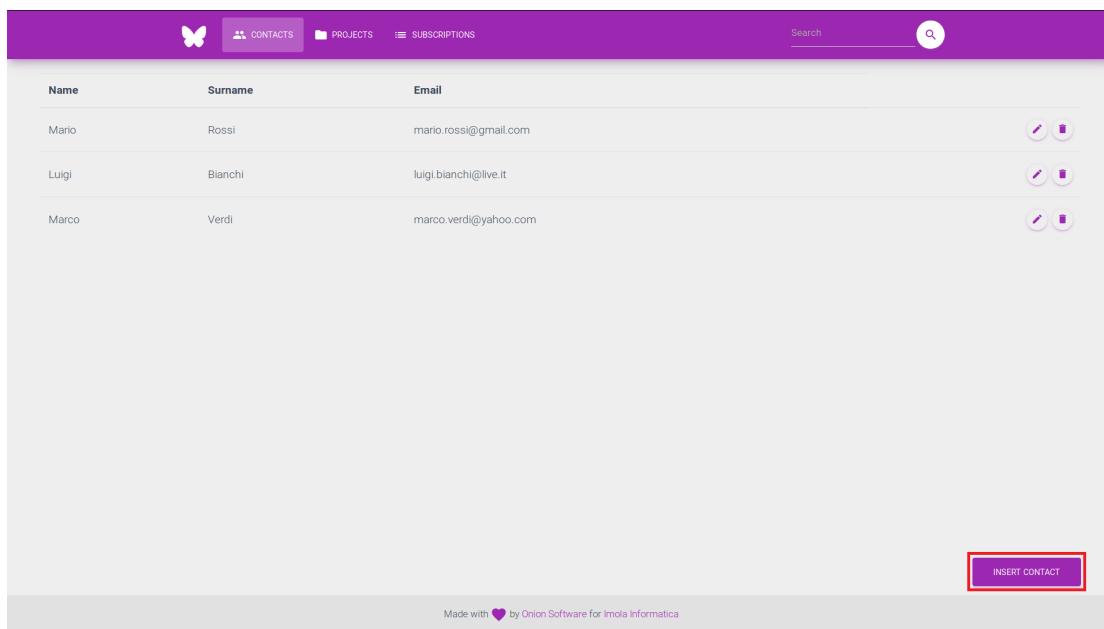


Figura 18: Pulsante INSERT CONTACT

2. si aprirà un *modal* in cui è possibile inserire le informazioni riguardanti il nuovo contatto da inserire;



The screenshot shows the Butterfly application interface. At the top, there's a purple header bar with the logo, navigation links for 'CONTACTS', 'PROJECTS', and 'SUBSCRIPTIONS', and a search bar. Below the header is a table listing three contacts: Mario Rossi, Luigi Bianchi, and Marco Verdi. A modal window titled 'INSERT CONTACT' is centered on the screen. Inside the modal, fields for 'Name' (Alberto), 'Surname' (Gialli), 'Email' (alberto.gialli@hotmail.it), 'Telegram username' (albertogialli), and 'Slack email' are present. There are date pickers for 'Holidays begin' (06/14/2019) and 'Holidays end' (06/21/2019). Under 'Notify on', the 'Email' and 'Slack' checkboxes are unchecked, while the 'Telegram' checkbox is checked. At the bottom of the modal are 'CLOSE' and 'SUBMIT' buttons.

Figura 19: *Modal* di inserimento contatto

3. confermare la creazione oppure annullare l'operazione;

This screenshot is similar to Figure 19, showing the 'INSERT CONTACT' modal. The contact information has been updated: Name is now Alberto and Surname is Gialli. The 'Email' field contains alberto.gialli@hotmail.it. The 'Telegram username' is albertogialli. The 'Holidays begin' date is 06/14/2019 and 'Holidays end' date is 06/21/2019. The 'Notify on' section shows that the 'Email' and 'Slack' checkboxes are unchecked, while the 'Telegram' checkbox is checked. The 'SUBMIT' button at the bottom of the modal is highlighted with a red rectangle.

Figura 20: Conferma di inserimento contatto

4. il nuovo contatto verrà visualizzato come ultimo elemento in lista.



The screenshot shows a list of contacts in the 'CONTACTS' tab. The table has columns for Name, Surname, and Email. Each contact row includes a pencil icon for editing and a trash bin icon for deleting. The contact 'Alberto Gialli' is highlighted with a gray background.

Name	Surname	Email
Mario	Rossi	mario.rossi@gmail.com
Luigi	Bianchi	luigi.bianchi@live.it
Marco	Verdi	marco.verdi@yahoo.com
Alberto	Gialli	alberto.gialli@hotmail.it

INSERT CONTACT

Made with ❤ by Onion Software for Imola Informatica

Figura 21: Visualizzazione contatto inserito

3.1.2.2.3 Modifica

Per modificare un contatto:

1. premere il pulsante di modifica situato a fianco del contatto che si vuole modificare;

The screenshot shows the same list of contacts, but the pencil icon next to the first contact ('Mario Rossi') is highlighted with a red box. This indicates it is the selected contact for modification.

Name	Surname	Email
Mario	Rossi	mario.rossi@gmail.com
Luigi	Bianchi	luigi.bianchi@live.it
Marco	Verdi	marco.verdi@yahoo.com

INSERT CONTACT

Made with ❤ by Onion Software for Imola Informatica

Figura 22: Pulsante di modifica contatto

2. si aprirà un *modal* in cui è possibile modificare le informazioni riguardanti il contatto selezionato;



Made with ❤ by Onion Software for Imola Informatica

Figura 23: *Modal* di modifica contatto

3. confermare la modifica oppure annullare l'operazione.

3.1.2.2.4 Eliminazione

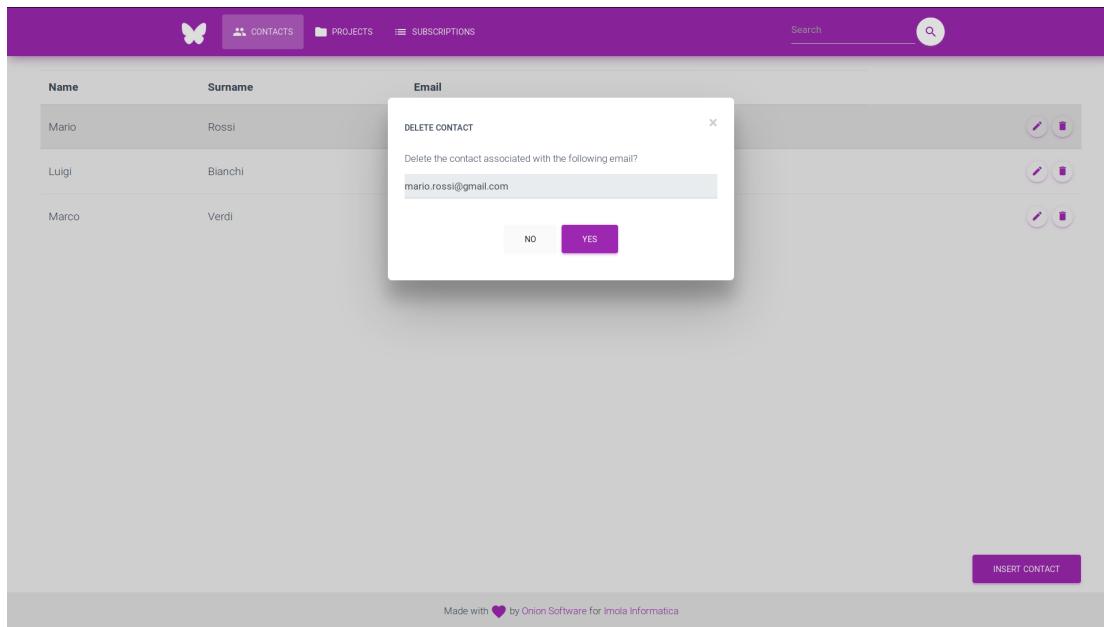
Per eliminare un contatto:

1. premere il pulsante di eliminazione situato a fianco del contatto che si vuole eliminare;

Made with ❤ by Onion Software for Imola Informatica

Figura 24: Pulsante di eliminazione contatto

2. si aprirà un *modal* in cui è possibile confermare l'eliminazione del contatto selezionato;

Figura 25: *Modal* di eliminazione contatto

3. confermare l'eliminazione oppure annullare l'operazione.

3.1.2.2.5 Ricerca

Per fare la ricerca di un contatto specifico:

1. scrivere una *keyword* nell'apposito box di ricerca situato in alto a destra e premere sulla relativa icona (oppure premere il pulsante INVIO);



Figura 26: Box di ricerca contatti

2. verrà visualizzata una lista di contatti che contengono tra le loro informazioni un match della *keyword* cercata, oppure un apposito messaggio nel caso in cui non venga trovata nessuna corrispondenza (fig. 17).

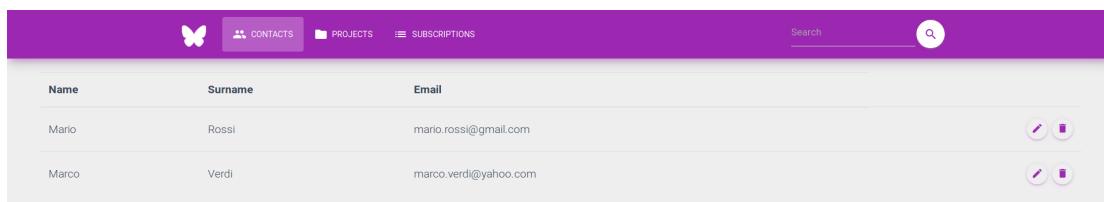


Figura 27: Visualizzazione contatti ricercati

3.1.2.2.6 Visualizzazione errori

Per ogni errore durante l'inserimento o la modifica di un contatto (come campi obbligatori non inseriti o non validi), verrà mostrato un messaggio che riporta la natura dell'errore.

Nell'esempio seguente si tenta di inserire un nuovo contatto con un'email che corrisponde già ad un contatto presente nel gestore personale.

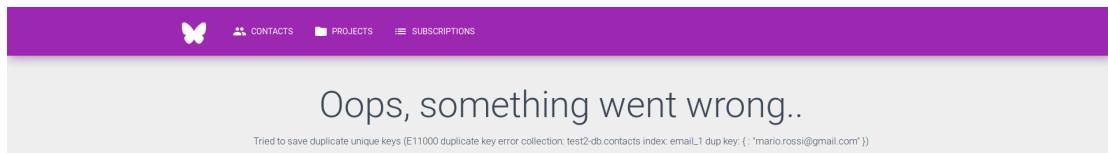


Figura 28: Visualizzazione messaggio d'errore

3.1.2.3 Progetti

3.1.2.3.1 Visualizzazione elementi

Questa sezione contiene la lista di tutti i progetti presenti all'interno del gestore personale.

Name	URL
Progetto 1	http://pro1.com/repo
Progetto 2	http://pro2.com/secret-repo

Figura 29: Sezione PROJECTS

Nel caso in cui non sia presente alcun progetto, verrà visualizzato un apposito messaggio.

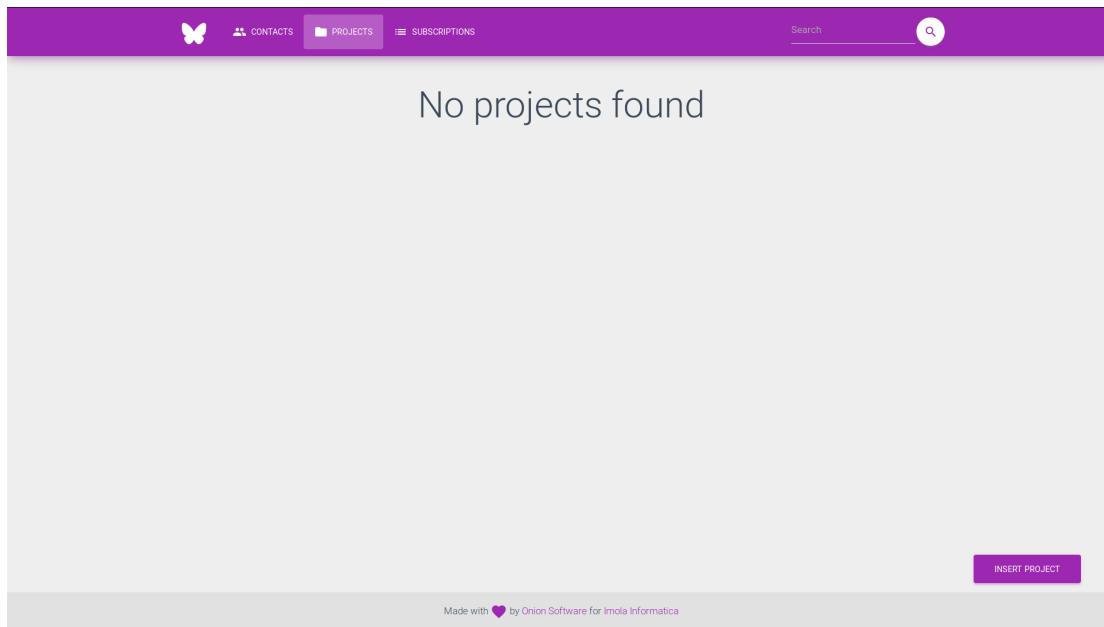
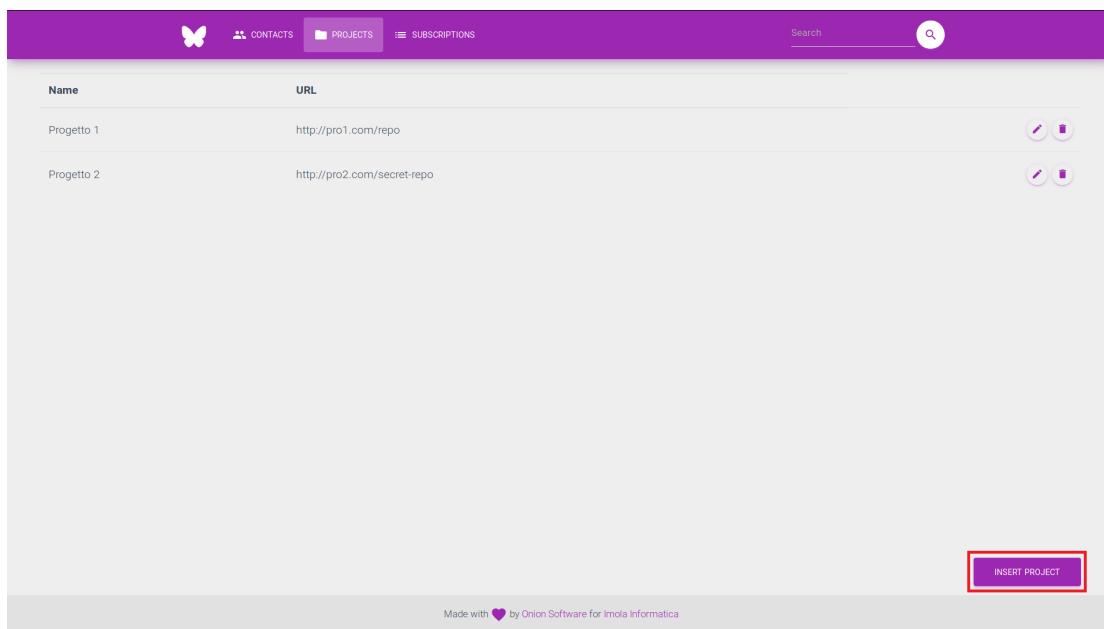


Figura 30: Sezione PROJECTS (vuota)

3.1.2.3.2 Inserimento

Per inserire un nuovo progetto:

1. premere il pulsante **INSERT PROJECT** situato in basso a destra;

Figura 31: Pulsante **INSERT PROJECT**

2. si aprirà un *modal* in cui è possibile inserire le informazioni riguardanti il nuovo progetto da inserire;

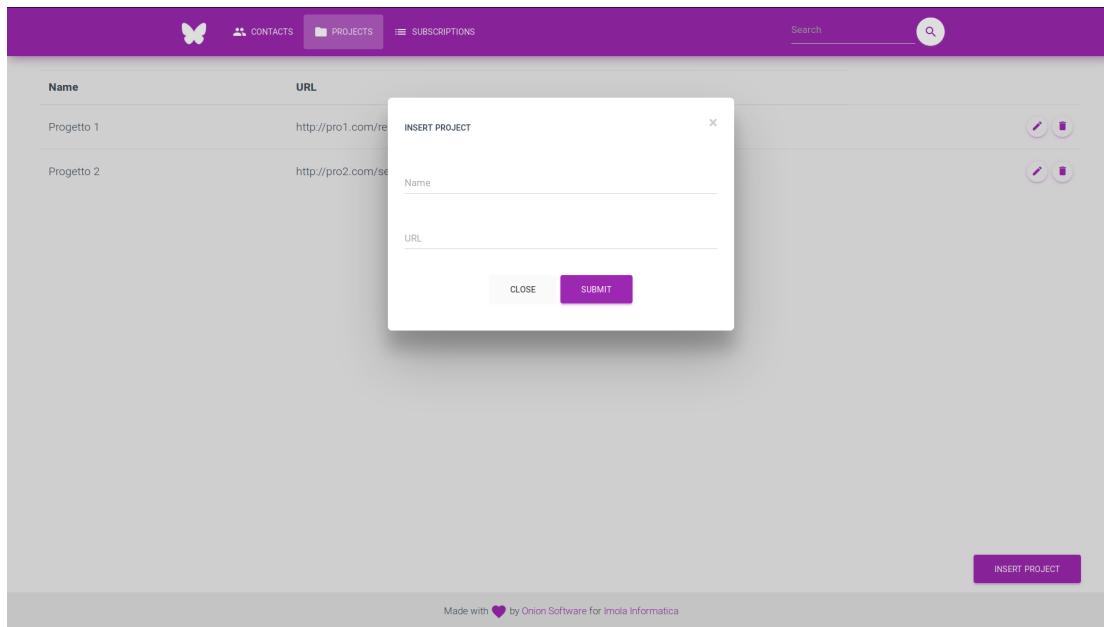


Figura 32: *Modal* di inserimento progetto

3. confermare la creazione oppure annullare l'operazione;

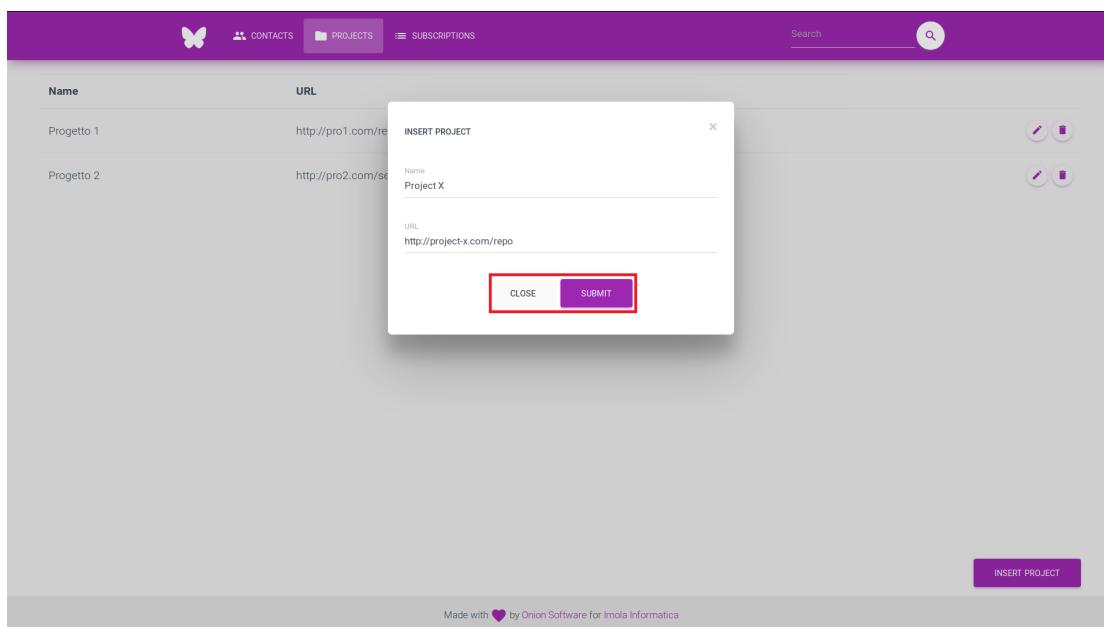


Figura 33: Conferma di inserimento progetto

4. il nuovo progetto verrà visualizzato come ultimo elemento in lista.



The screenshot shows the 'PROJECTS' tab selected in the top navigation bar. Below the header, there is a search bar and a purple 'INSERT PROJECT' button. The main area displays a table with three rows of project data:

Name	URL	
Progetto 1	http://pro1.com/repo	
Progetto 2	http://pro2.com/secret-repo	
Project X	http://project-x.com/repo	

At the bottom of the screen, a footer bar includes the text 'Made with ❤ by Onion Software for Imola Informatica'.

Figura 34: Visualizzazione progetto inserito

3.1.2.3.3 Modifica

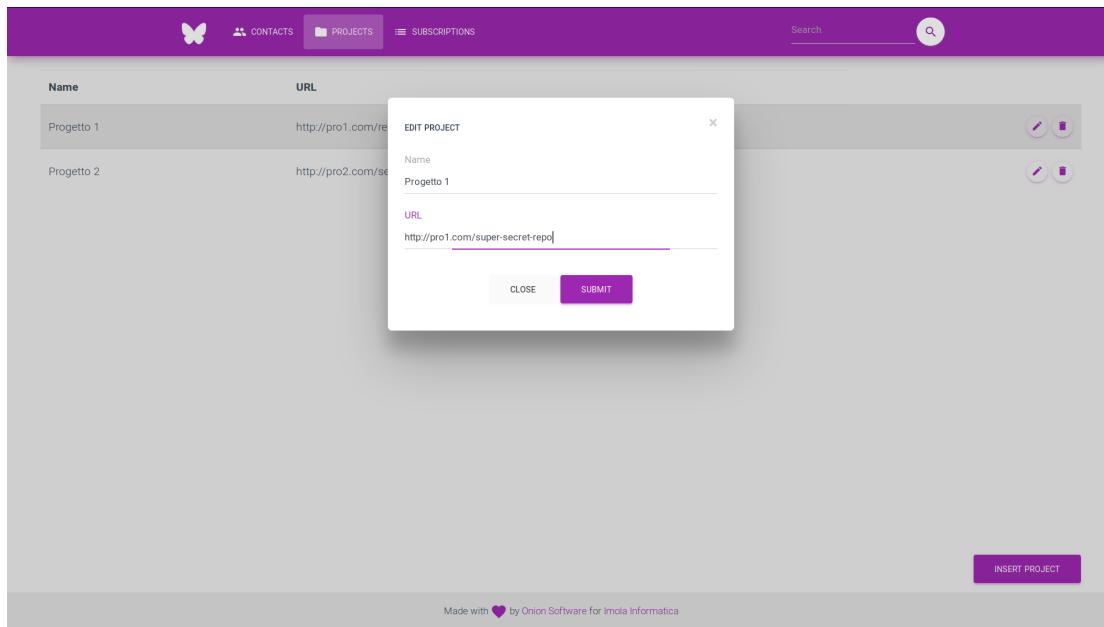
Per modificare un progetto:

1. premere il pulsante di modifica situato a fianco del progetto che si vuole modificare;

The screenshot is similar to Figure 34, showing the 'PROJECTS' tab selected. The 'Edit' icon for the 'Project X' row is highlighted with a red box. The rest of the interface, including the table and footer, remains the same.

Figura 35: Pulsante di modifica progetto

2. si aprirà un *modal* in cui è possibile modificare le informazioni riguardanti il progetto selezionato;

Figura 36: *Modal* di modifica progetto

3. confermare la modifica oppure annullare l'operazione.

3.1.2.3.4 Eliminazione

Per eliminare un progetto:

1. premere il pulsante di eliminazione situato a fianco del progetto che si vuole eliminare;

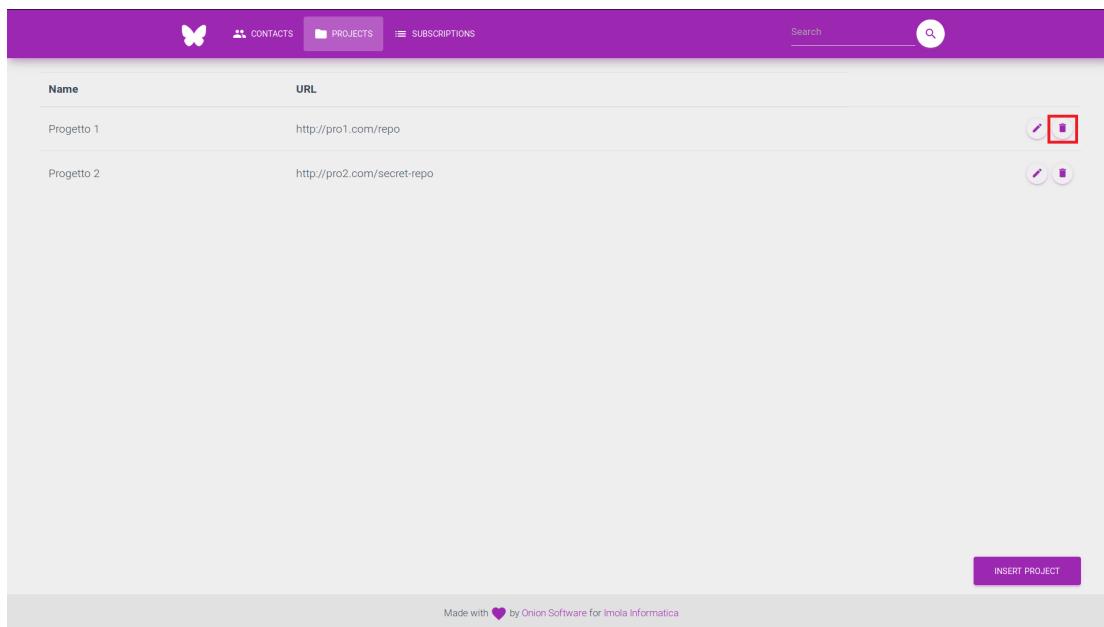
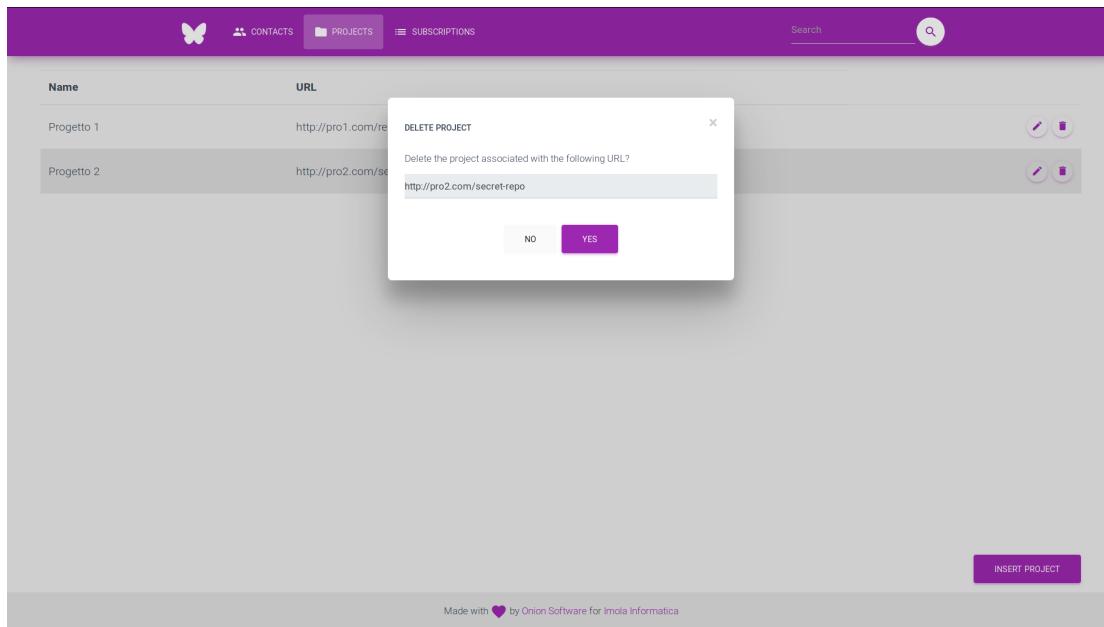


Figura 37: Pulsante di eliminazione progetto

2. si aprirà un *modal* in cui è possibile confermare l'eliminazione del progetto selezionato;

Figura 38: *Modal* di eliminazione progetto

3. confermare l'eliminazione oppure annullare l'operazione.

3.1.2.3.5 Ricerca

Per fare la ricerca di un progetto specifico:

1. scrivere una *keyword* nell'apposito box di ricerca situato in alto a destra e premere sulla relativa icona (oppure premere il pulsante INVIO);



Figura 39: Box di ricerca progetti

2. verrà visualizzata una lista di progetti che contengono tra le loro informazioni un match della *keyword* cercata, oppure un apposito messaggio nel caso in cui non venga trovata nessuna corrispondenza (fig. 30).

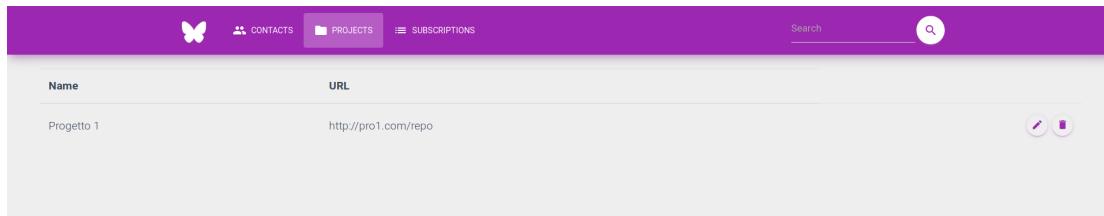


Figura 40: Visualizzazione progetti ricercati

3.1.2.3.6 Visualizzazione errori

Per ogni errore durante l'inserimento o la modifica di un progetto (come campi obbligatori non inseriti o non validi), verrà mostrato un messaggio che riporta la natura dell'errore.

Nell'esempio seguente si tenta di inserire un nuovo progetto con un URL che corrisponde già ad un progetto presente nel gestore personale.

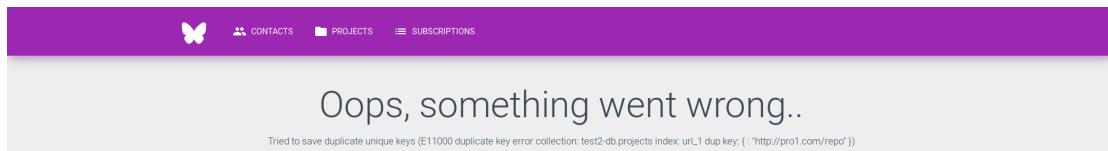


Figura 41: Visualizzazione messaggio d'errore

3.1.2.4 Iscrizioni

3.1.2.4.1 Visualizzazione elementi

Questa sezione contiene la lista di tutte le iscrizioni presenti all'interno del gestore personale.

Name	Surname	Project	URL	Priority	
Mario	Rossi	Progetto 1	http://pro1.com/repo	High	
Luigi	Bianchi	Progetto 2	http://pro2.com/secret-repo	Medium	
Mario	Rossi	Progetto 2	http://pro2.com/secret-repo	Low	

[INSERT SUBSCRIPTION](#)

Made with ❤ by Onion Software for Imola Informatica

Figura 42: Sezione SUBSCRIPTIONS

Nel caso in cui non sia presente alcuna iscrizione, verrà visualizzato un apposito messaggio.

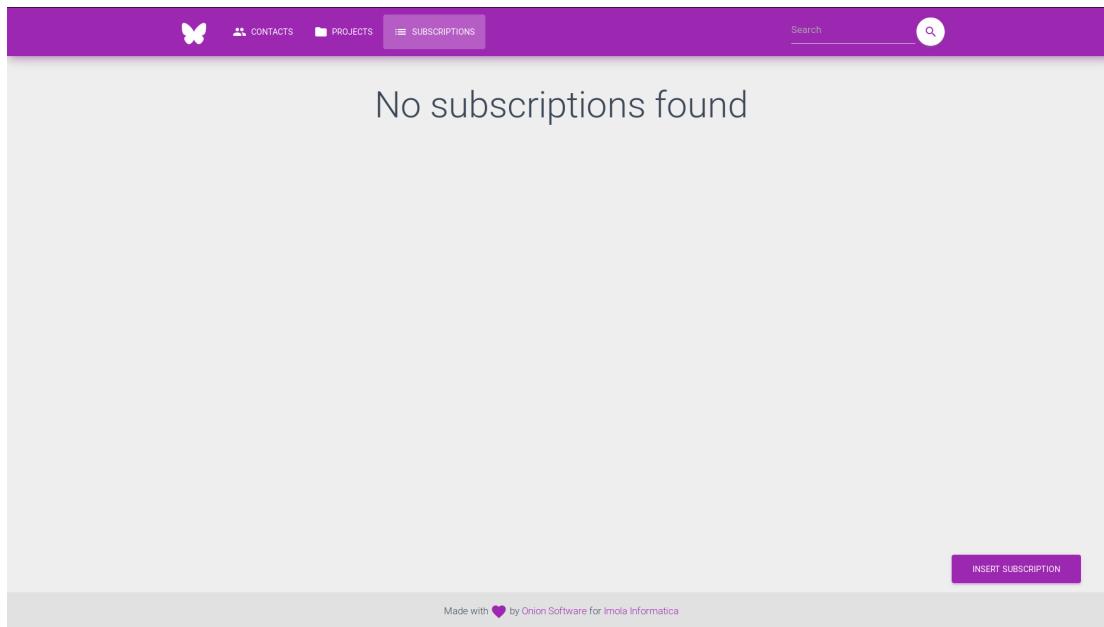


Figura 43: Sezione SUBSCRIPTIONS (vuota)

3.1.2.4.2 Inserimento

Per inserire una nuova iscrizione:

1. premere il pulsante **INSERT SUBSCRIPTION** situato in basso a destra;

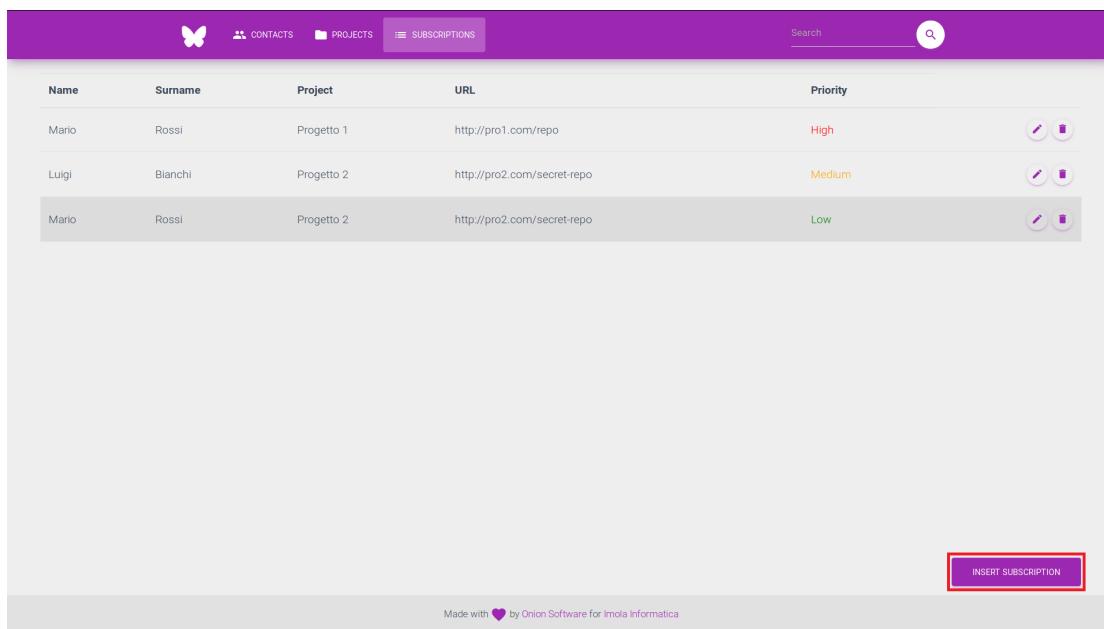


Figura 44: Pulsante INSERT SUBSCRIPTION

2. si aprirà un *modal* in cui è possibile inserire le informazioni riguardanti la nuova iscrizione da inserire;



Name	Surname	Project	URL	Priority
Mario	Rossi	Progetto 1	INSERT SUBSCRIPTION	High
Luigi	Bianchi	Progetto 2	Contact email luigi.bianchi@live.it	Medium
Mario	Rossi	Progetto 2	Project URL http://project-x.com/repo	Low

Priority
High

Keywords
frontend, devops

CLOSE SUBMIT

INSERT SUBSCRIPTION

Figura 45: *Modal* di inserimento iscrizione

3. confermare la creazione oppure annullare l'operazione;

Name	Surname	Project	URL	Priority
Mario	Rossi	Progetto 1	INSERT SUBSCRIPTION	High
Luigi	Bianchi	Progetto 2	Contact email luigi.bianchi@live.it	Medium
Mario	Rossi	Progetto 2	Project URL http://project-x.com/repo	Low

Priority
High

Keywords
frontend, devops

CLOSE SUBMIT

INSERT SUBSCRIPTION

Figura 46: Conferma di inserimento iscrizione

4. la nuova iscrizione verrà visualizzata come ultimo elemento in lista.



The screenshot shows a table with columns: Name, Surname, Project, URL, and Priority. The rows contain the following data:

Name	Surname	Project	URL	Priority
Mario	Rossi	Progetto 1	http://pro1.com/repo	High
Luigi	Bianchi	Progetto 2	http://pro2.com/secret-repo	Medium
Mario	Rossi	Progetto 2	http://pro2.com/secret-repo	Low
Luigi	Bianchi	Project X	http://project-x.com/repo	High

At the bottom right of the table area, there is a purple button labeled "INSERT SUBSCRIPTION". Below the table, a small note says "Made with ❤ by Onion Software for Imola Informatica".

Figura 47: Visualizzazione iscrizione inserita

3.1.2.4.3 Modifica

Per modificare un'iscrizione:

1. premere il pulsante di modifica situato a fianco dell'iscrizione che si vuole modificare;

The screenshot shows a table with columns: Name, Surname, Project, URL, and Priority. The rows contain the following data:

Name	Surname	Project	URL	Priority
Mario	Rossi	Progetto 1	http://pro1.com/repo	High
Luigi	Bianchi	Progetto 2	http://pro2.com/secret-repo	Medium
Mario	Rossi	Progetto 2	http://pro2.com/secret-repo	Low

In the third row, the edit icon (pencil icon) is highlighted with a red square. At the bottom right of the table area, there is a purple button labeled "INSERT SUBSCRIPTION". Below the table, a small note says "Made with ❤ by Onion Software for Imola Informatica".

Figura 48: Pulsante di modifica iscrizione

2. si aprirà un *modal* in cui è possibile modificare le informazioni riguardanti l'iscrizione selezionata;



The screenshot shows a list of subscriptions on the left and a detailed 'Edit Subscription' modal on the right. The modal contains fields for Contact email, Project URL, Priority, and Keywords. Buttons for CLOSE and SUBMIT are at the bottom.

Name	Surname	Project	URL	Priority
Mario	Rossi	Progetto 1		High
Luigi	Bianchi	Progetto 2		Medium
Mario	Rossi	Progetto 2		Low

Figura 49: *Modal* di modifica iscrizione

3. confermare la modifica oppure annullare l'operazione.

3.1.2.4.4 Eliminazione

Per eliminare un'iscrizione:

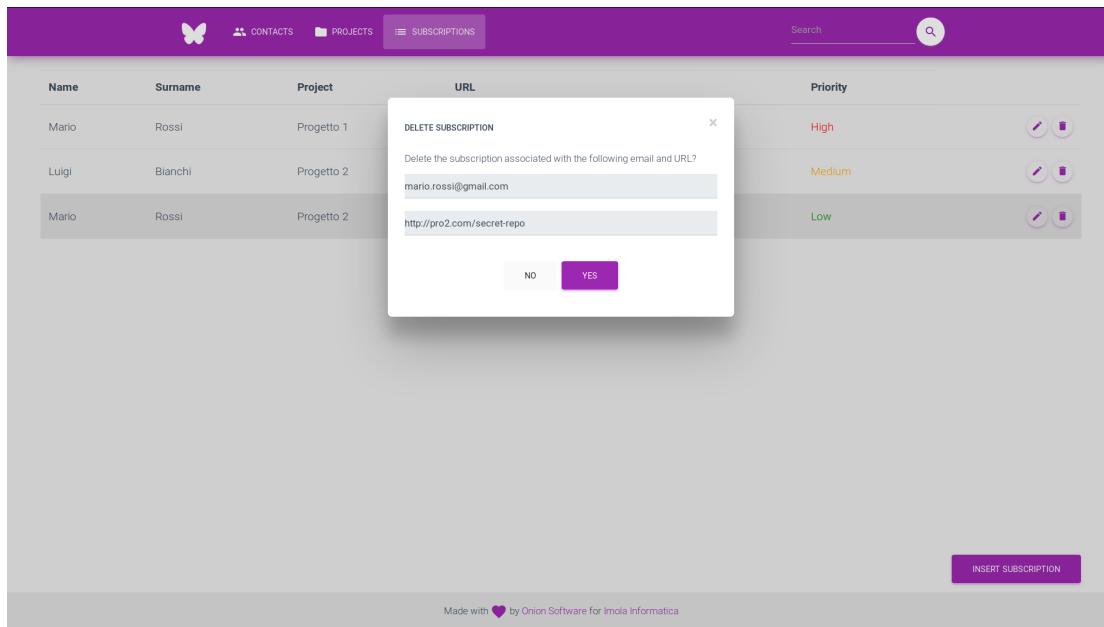
1. premere il pulsante di eliminazione situato a fianco dell'iscrizione che si vuole eliminare;

The screenshot shows a list of subscriptions. The third row (Mario, Rossi, Progetto 2) has a delete icon highlighted with a red box. The rest of the interface is identical to Figure 49.

Name	Surname	Project	URL	Priority
Mario	Rossi	Progetto 1	http://pro1.com/repo	High
Luigi	Bianchi	Progetto 2	http://pro2.com/secret-repo	Medium
Mario	Rossi	Progetto 2	http://pro2.com/secret-repo	Low

Figura 50: Pulsante di eliminazione iscrizione

2. si aprirà un *modal* in cui è possibile confermare l'eliminazione dell'iscrizione selezionata;

Figura 51: *Modal* di eliminazione iscrizione

3. confermare l'eliminazione oppure annullare l'operazione.

3.1.2.4.5 Ricerca

Per fare la ricerca di un'iscrizione specifica:

1. scrivere una *keyword* nell'apposito box di ricerca situato in alto a destra e premere sulla relativa icona (oppure premere il pulsante INVIO);



Figura 52: Box di ricerca iscrizioni

2. verrà visualizzata una lista di iscrizioni che contengono tra le loro informazioni un match della *keyword* cercata, oppure un apposito messaggio nel caso in cui non venga trovata nessuna corrispondenza (fig. 43).

Name	Surname	Project	URL	Priority	
Luigi	Bianchi	Progetto 2	http://pro2.com/secret-repo	Medium	
Mario	Rossi	Progetto 2	http://pro2.com/secret-repo	Low	

Figura 53: Visualizzazione iscrizioni ricercate

3.1.2.4.6 Visualizzazione errori

Per ogni errore durante l'inserimento o la modifica di un'iscrizione (come campi obbligatori non inseriti o non validi), verrà mostrato un messaggio che riporta la natura dell'errore. Nell'esempio seguente si tenta di inserire una nuova iscrizione con un'email che non è presente nel gestore personale.

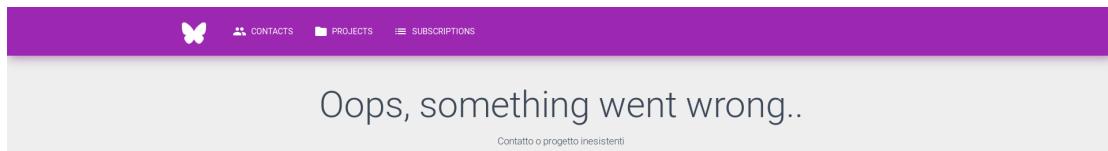


Figura 54: Visualizzazione messaggio d'errore

3.2 RESTful API

Sono state implementate delle API REST per la gestione delle risorse di Butterfly. Di seguito verrà spiegato come interagire con le API. Si noti che, per comodità, il *root path* G sottinteso sarà: $home_url/api/v1$.

3.2.1 Contatti

3.2.1.1 Lista

Per visualizzare la lista dei contatti è sufficiente effettuare la richiesta:

GET /contacts

```
[  
  {  
    "_id": {  
      "$oid": "5cfccfceb06db5f36afa19f97"  
    },  
    "name": "Linpeng",  
    "surname": "Zhang",  
    "email": "zhanglinpeng1998@gmail.com",  
    "telegram": {  
      "username": "Razionalmente"  
    },  
    "slack": {  
      "email": "zhanglinpeng1998@gmail.com"  
    },  
    "preferences": {  
      "use_email": true,  
      "use_telegram": true,  
      "use_slack": true  
    },  
    "holidays": {}  
  },  
  {  
    "_id": {  
      "$oid": "5cf01d58dc55ebd117a4d3f"  
    },  

```



```
        "use_email": true,
        "use_telegram": true,
        "use_slack": true
    },
    "holidays": {}
},
{
    "_id": {
        "$oid": "5cf02098dc55ebd117a4d40"
    },
    "name": "Federico",
    "surname": "Brian",
    "email": "federico.brian@gmail.com",
    "telegram": {
        "username": "bubogunz"
    },
    "slack": {
        "email": "federico.brian@gmail.com"
    },
    "preferences": {
        "use_email": true,
        "use_telegram": true,
        "use_slack": true
    },
    "holidays": {}
}
]
```

Listing 7: Esempio di output

3.2.1.2 Inserimento

Per inserire un contatto è sufficiente effettuare la richiesta:

POST /contacts

inserendo nel corpo dei dati in JSON le seguenti chiavi:

- contact_name*;
- contact_surname*;
- contact_email*;
- contact_telegram_username;
- contact_slack_email;
- contact_use_telegram;
- contact_use_slack;
- contact_use_email;
- contact_holidays_begin_1;
- contact_holidays_end_1;
- contact_holidays_begin_2;



- contact_holidays_end_2;
- contact_holidays_begin_3;
- contact_holidays_end_3.

dove i campi contrassegnati con il simbolo * sono obbligatori.

```
{  
    "contact_name": "Nicola",  
    "contact_surname": "Pastore",  
    "contact_email": "email@gmail.com",  
    "contact_telegram_username": "NicolaPastore",  
    "contact_slack_email": "ppastorking@gmail.com",  
    "contact_use_telegram": true,  
    "contact_use_slack": true,  
    "contact_use_email": true,  
    "contact_holidays_begin_1": "2019-06-07",  
    "contact_holidays_end_1": "2019-06-10"  
}
```

Listing 8: Esempio di input

```
{  
    "ok": "Contatto_inserito"  
}
```

Listing 9: Esempio di output

```
{  
    "errore": "Email_mancante"  
}
```

Listing 10: Esempio di output

3.2.1.3 Aggiornamento

Per aggiornare un contatto è sufficiente effettuare la richiesta:

PUT /contacts

inserendo nel corpo dei dati in JSON le seguenti chiavi:

- contact_email_old*;
- contact_name;
- contact_surname;
- contact_email;
- contact_telegram_username;
- contact_slack_email;
- contact_use_telegram;
- contact_use_slack;
- contact_use_email;
- contact_holidays_begin_1;



- contact_holidays_end_1;
- contact_holidays_begin_2;
- contact_holidays_end_2;
- contact_holidays_begin_3;
- contact_holidays_end_3;

dove i campi contrassegnati con il simbolo * sono obbligatori.

```
{  
    "contact_email_old": "email@gmail.com",  
    "contact_name": "Nicola",  
    "contact_surname": "Pastore",  
    "contact_email": "email@gmail.com",  
    "contact_telegram_username": "NicolaPastore",  
    "contact_slack_email": "ppastorking@gmail.com",  
    "contact_use_telegram": true,  
    "contact_use_slack": true,  
    "contact_use_email": true,  
    "contact_holidays_begin_1": "2019-06-07",  
    "contact_holidays_end_1": "2019-06-10"  
}
```

Listing 11: Esempio di input

```
{  
    "ok": "Contatto_aggiornato"  
}
```

Listing 12: Esempio di output

```
{  
    "errore": "Email_del_contatto_da_modificare_mancante"  
}
```

Listing 13: Esempio di output

```
{  
    "errore": "Contatto_da_modificare_inesistente"  
}
```

Listing 14: Esempio di output

3.2.1.4 Cancellazione

Per eliminare un contatto è sufficiente effettuare la richiesta:

DELETE /contacts

inserendo nel corpo dei dati in JSON le seguenti chiavi:

- email_to_delete*.

dove i campi contrassegnati con il simbolo * sono obbligatori.



```
{  
    "email_to_delete": "test@test.com"  
}
```

Listing 15: Esempio di input

```
{  
    "ok": "Contatto_eliminato"  
}
```

Listing 16: Esempio di output

```
{  
    "errore": "Email_mancante"  
}
```

Listing 17: Esempio di output

```
{  
    "errore": "Contatto_inesistente"  
}
```

Listing 18: Esempio di output

3.2.2 Progetti

3.2.2.1 Lista

Per visualizzare la lista dei progetti è sufficiente effettuare la richiesta:

GET /projects

```
[  
    {  
        "_id": {  
            "$oid": "5cf03498dc55ebd117a4d41"  
        },  
        "name": "Test_Project",  
        "url": "https://gitlab.com/linpengzhang/test"  
    },  
    {  
        "_id": {  
            "$oid": "5cg03498dc43dbd118a4d41"  
        },  
        "name": "Test2_Project",  
        "url": "https://gitlab.com/linpengzhang/test2"  
    }  
]
```

Listing 19: Esempio di output

3.2.2.2 Inserimento

Per inserire un progetto è sufficiente effettuare la richiesta:

POST /projects



inserendo nel corpo dei dati in JSON le seguenti chiavi:

- project_name*;
- project_url*.

dove i campi contrassegnati con il simbolo * sono obbligatori.

```
{  
    "project_name": "Test_Project",  
    "project_url": "https://gitlab.com/linpengzhang/test"  
}
```

Listing 20: Esempio di input

```
{  
    "ok": "Progetto_inserito"  
}
```

Listing 21: Esempio di output

```
{  
    "errore": "Nome_mancante"  
}
```

Listing 22: Esempio di output

```
{  
    "errore": "Url_mancante"  
}
```

Listing 23: Esempio di output

3.2.2.3 Aggiornamento

Per inserire un'iscrizione è sufficiente effettuare la richiesta:

PUT /projects

inserendo nel corpo dei dati in JSON le seguenti chiavi:

- project_url_old*.
- project_name;
- project_url.

dove i campi contrassegnati con il simbolo * sono obbligatori.

```
{  
    "project_url_old": "https://gitlab.com/linpengzhang/test",  
    "project_name": "New_project_name",  
    "project_url": "https://gitlab.com/linpengzhang/test"  
}
```

Listing 24: Esempio di input

```
{  
    "ok": "Progetto_aggiornato"  
}
```

Listing 25: Esempio di output



```
{  
    "errore": "Url del progetto da modificare mancante"  
}
```

Listing 26: Esempio di output

```
{  
    "errore": "Progetto inesistente"  
}
```

Listing 27: Esempio di output

3.2.2.4 Cancellazione

Per eliminare un progetto è sufficiente effettuare la richiesta:

DELETE /projects

inserendo nel corpo dei dati in JSON le seguenti chiavi:

- url_to_delete*.

dove i campi contrassegnati con il simbolo * sono obbligatori.

```
{  
    "url_to_delete": "https://gitlab.com/linpengzhang/test"  
}
```

Listing 28: Esempio di input

```
{  
    "ok": "Progetto eliminato"  
}
```

Listing 29: Esempio di output

```
{  
    "errore": "Url mancante"  
}
```

Listing 30: Esempio di output

```
{  
    "errore": "Progetto inesistente"  
}
```

Listing 31: Esempio di output

3.2.3 Iscrizioni

3.2.3.1 Lista

Per visualizzare la lista delle iscrizioni è sufficiente effettuare la richiesta:

GET /subscriptions



```
[  
  {  
    "_id": {  
      "$oid": "5cf035a8dc55ebd117a4d42"  
    },  
    "contact": {  
      "$oid": "5cfcfceb06db5f36afa19f97"  
    },  
    "project": {  
      "$oid": "5cf03498dc55ebd117a4d41"  
    },  
    "priority": 3,  
    "keywords": [  
      "test"  
    ]  
  }  
]
```

Listing 32: Esempio di output

3.2.3.2 Inserimento

Per inserire un'iscrizione è sufficiente effettuare la richiesta:

POST /subscriptions

inserendo nel corpo dei dati in JSON le seguenti chiavi:

- contact_email*;
- project_url*;
- subscription_priority;
- subscription_keywords;

dove i campi contrassegnati con il simbolo * sono obbligatori.

```
{  
  "contact_email": "zhanglinpeng1998@gmail.com",  
  "project_url": "https://gitlab.com/linpengzhang/test",  
  "subscription_priority": "3",  
  "subscription_keywords": "test , progetto , parola",  
}
```

Listing 33: Esempio di input

```
{  
  "ok": "Iscrizione inserita"  
}
```

Listing 34: Esempio di output

```
{  
  "errore": "Email mancante"  
}
```

Listing 35: Esempio di output



```
{  
    "errore": "Url_mancante"  
}
```

Listing 36: Esempio di output

3.2.3.3 Aggiornamento

Per aggiornare un'iscrizione è sufficiente effettuare la richiesta:

PUT /subscriptions

inserendo nel corpo dei dati in JSON le seguenti chiavi:

- contact_email_old*;
- project_url_old*;
- contact_email;
- project_url;
- subscription_priority;
- subscription_keywords;

dove i campi contrassegnati con il simbolo * sono obbligatori.

```
{  
    "contact_email_old": "zhanglinpeng1998@gmail.com",  
    "project_url_old": "https://gitlab.com/linpengzhang/test",  
    "contact_email": "zhanglinpeng1998@gmail.com",  
    "project_url": "https://gitlab.com/linpengzhang/test",  
    "subscription_priority": "2",  
    "subscription_keywords": "test,progetto,nuova parola",  
}
```

Listing 37: Esempio di input

```
{  
    "ok": "Iscrizione_aggiornata"  
}
```

Listing 38: Esempio di output

```
{  
    "errore": "Email_o_url_dell'iscrizione_mancanti"  
}
```

Listing 39: Esempio di output

```
{  
    "errore": "Iscrizione_inesistente"  
}
```

Listing 40: Esempio di output



3.2.3.4 Cancellazione

Per eliminare un'iscrizione è sufficiente effettuare la richiesta:

```
DELETE /subscriptions
```

inserendo nel corpo dei dati in JSON le seguenti chiavi:

- email_to_delete*;
- url_to_delete*.

dove i campi contrassegnati con il simbolo * sono obbligatori.

```
{  
    "email_to_delete": "zhanglinpeng1998@gmail.com"  
    "url_to_delete": "https://gitlab.com/linpengzhang/test"  
}
```

Listing 41: Esempio di input

```
{  
    "ok": "Iscrizione_eliminata"  
}
```

Listing 42: Esempio di output

```
{  
    "errore": "Url_mancante"  
}
```

Listing 43: Esempio di output

```
{  
    "errore": "Email_mancante"  
}
```

Listing 44: Esempio di output

```
{  
    "errore": "Iscrizione_inesistente"  
}
```

Listing 45: Esempio di output

3.3 Ricezione notifiche

Riportiamo di seguito un esempio di notifica inviata da ciascuno dei nostri *producer*, con eventuali informazioni aggiuntive a riguardo:

- email:



Figura 55: Esempio di notifica email

- Telegram: per ricevere le notifiche provenienti da Telegram è necessario dare il consenso al bot @butterfly_bot l'invio dei messaggi verso il proprio account avviando semplicemente la conversazione o tramite il comando /start. L'esempio di notifica è il seguente:

Hi there!
We would like to inform you that user Linpeng Zhang submitted a new issue on project: "Test project 1" (link: <http://localhost:4080/linpengzhang/test-project-1>) via GitLab.
Here's what's in the notification:
Author: Linpeng Zhang
Title: Issue di prova
Issue URL: <http://gitlab.server/linpengzhang/test-project-1/issues/5>
Status: closed
Assignee: linpengzhang
- Created on: 2019-07-14 09:34:11 UTC
- Description: Descrizione di prova

11:50

Figura 56: Esempio di notifica Telegram

- Slack: per ricevere le notifiche provenienti da Slack è necessario aggiungere il bot *Butterfly* nel proprio workspace. L'esempio di notifica è il seguente:

Butterfly APP 11:50 AM
Hi there!
We would like to inform you that user Linpeng Zhang submitted a new issue on project: "Test project 1" (link: <http://localhost:4080/linpengzhang/test-project-1>) via GitLab.
Here's what's in the notification:
Author: Linpeng Zhang
Title: Issue di prova
Issue URL: <http://gitlab.server/linpengzhang/test-project-1/issues/5>
Status: closed
Assignee: linpengzhang
- Created on: 2019-07-14 09:34:11 UTC
- Description: Descrizione di prova

Figura 57: Esempio di notifica Slack



3.4 Nota sulla ricezione delle notifiche

Come concordato con l’azienda proponente *Imola Informatica S.p.A.*, la ricezione delle notifiche viene gestita in due “strati”:

- priorità dell’iscrizione (alta, media, bassa);
- *keyword* utilizzate.

Per ogni notifica che deve essere inviata, viene fatto un *matching* di *keyword* tra i contatti che sono iscritti a quel progetto, dando precedenza a quelli con priorità più alta e che sono disponibili (non in vacanza) a riceverla.

Nel caso della priorità più alta, non avere *keyword* inserite corrisponde ad avere un “jolly” per il match, di conseguenza le notifiche verranno inviate solamente a quelli che **non** hanno alcuna parola chiave e a tutti quelli che hanno quella parola tra le *keyword*.

Nel caso delle altre due priorità, la notifica viene inviata esclusivamente a chi possiede quella parola.



4 Contatti

Di seguito sono riportate le modalità con cui è possibile contattare il team *Onion Software*, responsabile dello sviluppo di *Butterfly*.

4.1 Segnalazioni e suggerimenti

4.1.1 Github

Il metodo di comunicazione preferenziale per le segnalazioni strettamente riguardanti il software *Butterfly*, come ad esempio il riscontro di bug o incompatibilità, è la pagina dedicata alle **issue** all'interno del repository pubblico di Github, disponibile all'indirizzo <https://github.com/Onion-Software/Butterfly>.

Affinché l'intervento degli sviluppatori sia mirato e tempestivo è importante che la segnalazione venga effettuata seguendo alcuni accorgimenti:

1. **Titolo:** il titolo è parte fondamentale della segnalazione, deve essere un buon compromesso tra concisione ed esaustività;
2. **Labels:** le labels, insieme al titolo della issue, danno le prime informazioni utili a capire tipologia ed entità della segnalazione. È importante quindi sceglierle in modo oculato tra queste possibilità predefinite: ENHANCEMENT, FIX o BUG;
3. **Commento o descrizione:** in questa sezione è buona norma seguire queste indicazioni:
 - lasciare un indice di priorità che si ritiene adeguato alla segnalazione in atto, compresa tra ALTA, MEDIA o BASSA;
 - scrivere una descrizione completa ed esaustiva del fatto scatenante la segnalazione;
 - descrivere dettagliatamente le condizioni in cui è avvenuto il fatto e l'ambiente di esecuzione in cui si è verificato;
 - fare uso, quando necessario o chiarificatore, di risorse come screenshots, elenchi puntati e formattazione del testo in *markdown*.
4. **Conferma:** ricordarsi di fare click sul pulsante indicato al punto 4 della figura per confermare l'inserimento della issue.

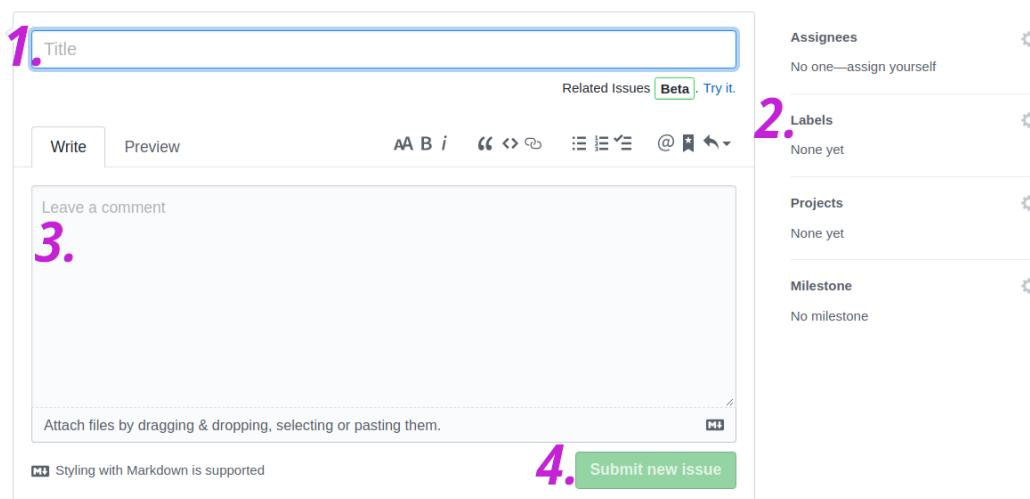


Figura 58: Inserimento di una issue su Github



4.1.2 Email

L'uso dell'email è consigliato per le segnalazioni di carattere generico o per comunicazioni di altra natura rispetto ai problemi riscontrati con il software *Butterfly*. Per segnalazioni di bug o problematiche legate al software è consigliabile seguire le indicazioni riportate in sezione §4.1.1 di questo manuale.

Il team *Onion Software* chiede cortesemente di attenersi a queste linee guida per le email ad esso indirizzate:

- scrivere chiaramente lo scopo della mail nel campo *oggetto* della stessa;
- essere chiari e concisi nel corpo della mail, in caso di richieste numerose o particolarmente articolate è apprezzato l'uso di elenchi puntati e formattazione adeguata del testo;
- firmarsi e lasciare, ove possibile, un recapito alternativo rispetto all'indirizzo con cui si è scritta la mail in questione (ad esempio un numero telefonico).

4.2 Contribuire allo sviluppo

Se si vuole contribuire allo sviluppo di *Butterfly*, si può fare un fork del repository pubblico del progetto open source su Github al seguente indirizzo:

<https://github.com/Onion-Software/Butterfly>. Successivamente è sufficiente, come avviene per ogni progetto open source, apportare le modifiche desiderate al fork del progetto *Butterfly* e, una volta ritenute stabili, aprire una *pull request* affinché vengano revisionate e accettate nel flusso principale di sviluppo.

Un'altra risorsa utile per poter contribuire all'applicativo è il documento denominato *manualeSviluppatore*, il quale contiene informazioni utili alla comprensione dell'architettura e del codice ad oggi esistente di *Butterfly*.



A Glossario

C

Container

Un container è una struttura software che rende l'esecuzione degli applicativi al suo interno indipendenti dall'ambiente di esecuzione del container. Più container possono coesistere sulla stessa macchina in modo indipendente tra loro, evitando l'installazione e la manutenzione di una macchina virtuale complessa.

D

Docker-Compose

Docker-compose è uno strumento per definire e lanciare applicazioni suddivise su più container. Con Compose si utilizza un file in formato YAML per configurare i servizi dell'applicativo, affinché poi sia possibile creare e lanciare tutti i servizi configurati in precedenza con un singolo comando.

Dockerfile

Il dockerfile è un documento di testo che contiene tutti i comandi necessari ad istanziare e configurare dei container Docker.

M

Markdown

Markdown è un linguaggio di markup con una sintassi del testo semplice, progettata in modo che possa essere facilmente convertita in HTML e in molti altri formati usando un tool omonimo.

R

Rancher

Rancher è uno strumento di gestione centralizzata di cluster Kubernetes che mira a semplificarne l'organizzazione.

W

Webhook

Un Webhook (in italiano letteralmente: "uncino del web") in programmazione web è un metodo per aumentare o alterare il comportamento di una pagina o applicazione web con chiamate di ritorno personalizzate. Queste chiamate di ritorno possono essere mantenute, modificate e gestite da utenti di terze parti e chi le sviluppa non fa necessariamente parte del sito o applicazione d'origine.