



MDI: Manuale Di Installazione

Riferimento	
Versione	2.0
Data	25/01/2022
Destinatario	Prof. Carmine Gravino
Presentato da	Danilo Aliberti
Approvato da	



Revision history

Data	Versione	Descrizione	Autore
24/01/2022	1.0	Stesura iniziale	Danilo Aliberti
25/01/2022	2.0	Revisione e aggiornamento	Danilo Aliberti



Sommario

REVISION HISTORY	1
SOMMARIO	2
1. INTRODUZIONE	3
1.1 SCOPO DEL SISTEMA	3
1.2 SCOPO DEL DOCUMENTO	3
1.3 RIFERIMENTI	3
2. PREREQUISITI PER L'INSTALLAZIONE	4
2.1 APPLICAZIONE WEB	4
3. INSTALLAZIONE IN LOCALE E CONFIGURAZIONE	4
3.1 CREAZIONE AMMINISTRATORE DJANGO	5
3.2 PRIMO ACCESSO ALL'AMMINISTRAZIONE	5
3.3 CONFIGURAZIONE "SITI"	7
3.4 CONFIGURAZIONE TOKEN DI ACCESSO SOCIAL	8
3.5 CONFIGURAZIONE PROFILI	10
3.6 CONFIGURAZIONE E-MAIL	11
4. GLOSSARIO	12



1. Introduzione

1.1 Scopo del sistema

Il sistema che si vuole realizzare parte dall'idea di poter ampliare e migliorare una già esistente piattaforma di gestione del Fantacalcio.

BiaSet nasce per poter offrire un aiuto ai gestori di campionati fantacalcistici, i quali preferiscono le regole particolari da noi create per movimentare il classico e noioso Fantacalcio.

Utilizzando la nostra piattaforma, il gestore del campionato non dovrà più preoccuparsi dei processi più complessi della gestione di un campionato, quali, ad esempio, la creazione del calendario di scontri, il calcolo degli stipendi dei giocatori e il calcolo automatizzato di fine giornata.

Allo stesso modo, sarà più semplice per gli allenatori gestire la propria squadra all'interno del sistema. Essi potranno anche comunicare tra loro attraverso un servizio di messaggistica.

Il sistema è gestito da un League Admin, il quale ha pieno controllo su tutti i campionati e gli utenti. Tale utente avrà accesso esclusivo, oltre che alla dashboard, ad un pannello di amministrazione privilegiato, fornito dal framework utilizzato.

1.2 Scopo del documento

Lo scopo del presente documento è quello di introdurre il manutentore ai primi passi, a partire dall'installazione del sistema.

Oltre a ciò, è presente un ulteriore capitolo che spiega come configurare i token dei social network per i servizi base di registrazione e login.

1.3 Riferimenti

Il presente documento è relazionato ad altri precedentemente stilati. Di seguito una lista:

- Problem Statement;
- Requirement Analysis Document;
- System Design Document;
- Object Design Document;
- Test Plan;
- Matrice di tracciabilità.



2. Prerequisiti per l'installazione

I prerequisiti necessari per l'installazione di Biaset sono:

- Un server con qualsiasi sistema operativo installato che sia in grado di supportare **Docker**;
- **Docker** installato sulla macchina;
- **Docker Compose** installato;
- **Git**.

2.1 Applicazione web

Biaset è un'applicazione web che raggruppa diversi tipi di files: classi Python, HTML, CSS, JS e altre risorse che la compongono.

L'applicazione fa uso di un database relazionale MySQL per la gestione dei dati persistenti. La connessione al database e le interazioni con esso sono interamente gestite dal framework Django, attraverso il suo ORM. Nelle fasi embrionali di sviluppo è stato utilizzato un database su file, precisamente SQLite.

3. Installazione in locale e configurazione

Come precedentemente indicato, la macchina sulla quale dovrà essere installata l'applicazione dovrà necessariamente avere Docker e il suo compose preinstallati.

Si preleva il codice da Github, utilizzando Git via shell sulla macchina interessata. Ci si posiziona nella cartella in cui sono presenti i files *Dockerfile* e *docker-compose.yaml* e si esegue il seguente comando:

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

L'installazione partirà automaticamente. Docker penserà a scaricare e installare tutte le dipendenze richieste dalla web app. Una volta completato il tutto, Docker avrà creato due containers:

- *Biaset_container*, il quale conterrà la nostra web app;
- *Mysql_biaset_db*, che conterrà il database MySQL.

N.B.: è possibile che alla prima installazione il container di Django non venga inizializzato correttamente. Riavviare il container corregge il problema.

Aprire un'altra finestra shell al termine.



3.1 Creazione amministratore Django

Questo passo è di fondamentale importanza, in quanto è necessaria la creazione del primo utente che avrà accesso alla piattaforma.

Si dovrà accedere alla shell del container *biaset_container*. Per fare ciò, sarà necessario digitare nella shell del server:

```
docker exec -it biaset_container /bin/bash
```

Una volta avuto accesso alla shell del container, ci si dovrà posizionare nella cartella contenente il file *manage.py*, ovvero */app/biaset/*.

A questo punto si dovrà digitare il comando:

```
python manage.py createsuperuser
```

Seguendo i passi (inserimento username, indirizzo email, password, conferma password), si procederà alla creazione del primo utente amministratore del sistema.

```
root@7871bb35b841:/app/biaset# ./manage.py createsuperuser
Nome utente (leave blank to use 'root'): freterica
Indirizzo email: freterica@gmail.com
Password:
Password (again):
Superuser created successfully.
```

3.2 Primo accesso all'amministrazione

Il framework Django mette a disposizione una vasta gamma di funzionalità di gestione di tutta la piattaforma, tramite l'accesso alla pagina di amministrazione, raggiungibile all'indirizzo <http://indirizzo-macchina/admin> (nel nostro caso, <http://localhost:8000/admin/>).

Una volta aperta la pagina, inserire i dati dell'amministratore precedentemente creato ed effettuare l'accesso.



BiaSet Management

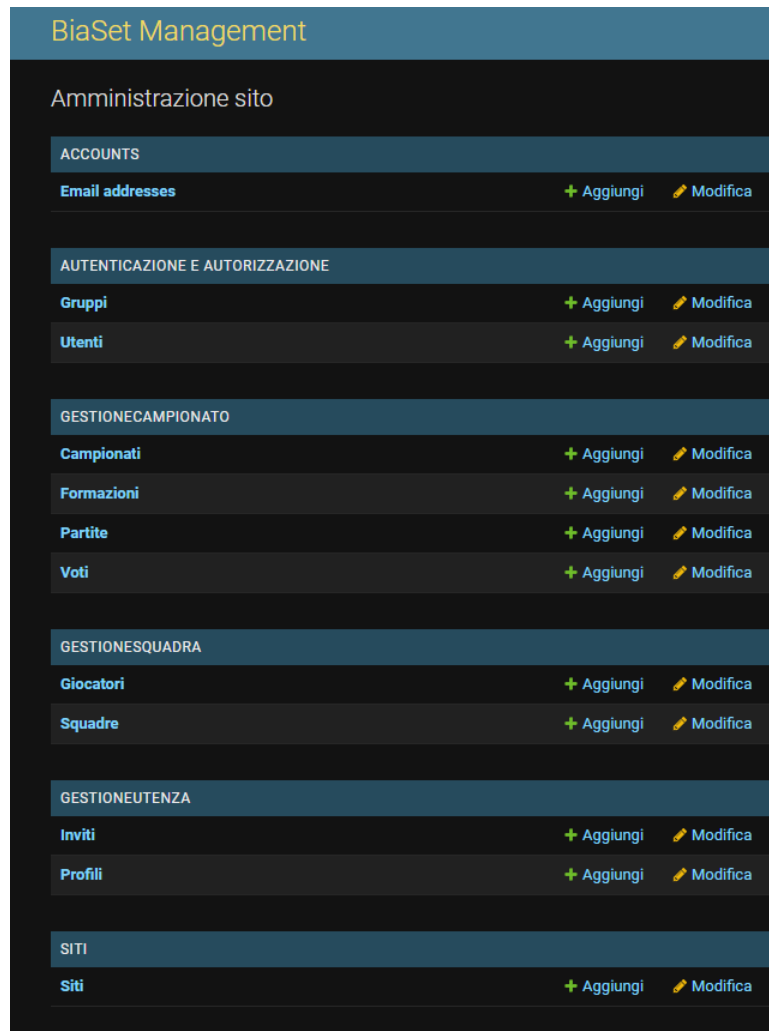
Nome utente:

freterica

Password:

Accedi

Una volta effettuato l'accesso, ci troveremo di fronte questa pagina:



Congratulazioni! Hai appena effettuato il primo accesso alla piattaforma. Ora procediamo alla configurazione base del sistema.

3.3 Configurazione “Siti”

Una volta loggato, sarà necessario configurare l'indirizzo del sito web, affinché in tutta la piattaforma i links siano corretti.

Nella pagina di amministrazione, cliccare su “Siti” e, successivamente, sull'unico record presente, ovvero “example” e modificarlo inserendo i dati “localhost:8000” e “localhost” rispettivamente nei campi “Nome di dominio” e “Nome visualizzato” e cliccare su “Salva”:

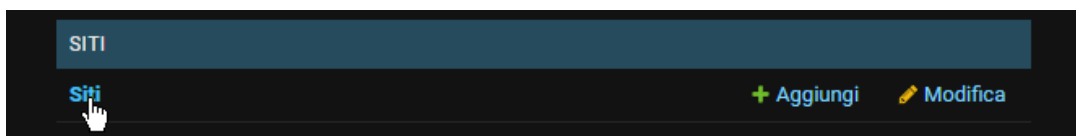


Figura 1 Clicca su "Siti"

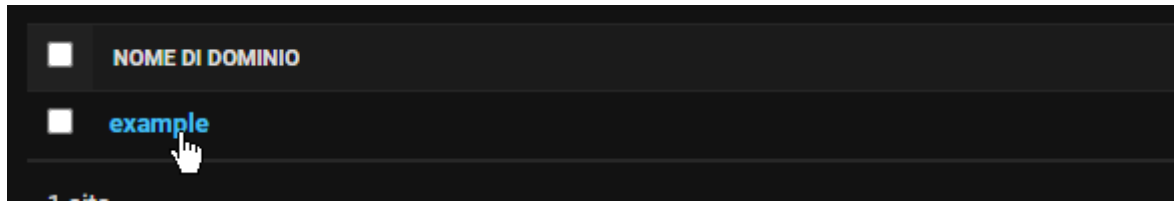


Figura 2 Seleziona "example"

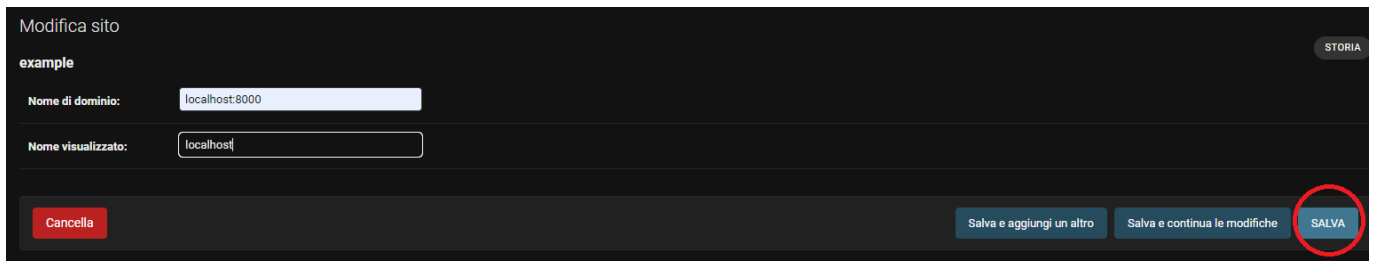


Figura 3 Imposta i dati e salva

Ritornare nella home page della pagina di amministrazione cliccando su “Biaset management” in alto a sinistra.

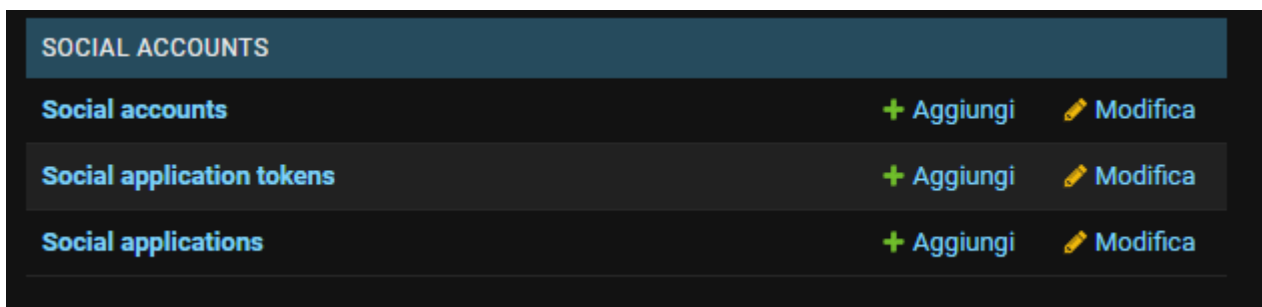
3.4 Configurazione Token di Accesso Social

Attenzione: prima di effettuare questa configurazione, sarà necessario possedere delle chiavi API su **Google Cloud Console**. La creazione si effettua su <https://console.developers.google.com/> , creando un nuovo progetto e generando le Credenziali (ID Client OAuth) dalla dashboard. Sarà necessario generare una chiave API anche per quanto riguarda **Facebook**. Si rimanda a questo [link](#) per la configurazione (fino allo step 4).

N.B.: in entrambe le configurazioni di accesso dovranno essere specificati gli indirizzi autorizzati per le **callbacks**. Nel caso di **Google**, l'indirizzo da aggiungere sarà <http://localhost:8000/accounts/google/login/callback/> .

Per **Facebook**, invece, <http://127.0.0.1:8000/accounts/google/login/callback/> .

Una volta sulla home di amministrazione, recarsi nella sezione relativa ai “Social Accounts” e cliccare su **Social Applications**.



Cliccare sul tasto “Aggiungi Social Application”, compilando il form come segue:

- **Provider:** selezionate Google o Facebook;
- **Nome:** un nome da dare al provider. Esempio: Facebook Provider / Google provider
- **Client ID:** corrisponde all’ID Client che vi è stato fornito dalle credenziali generate su Google Cloud Platform o sul portale di Facebook per developers.
 - Per **Google**, tale informazione si troverà all’interno della sezione “Credenziali”, ID Client OAuth 2.0;

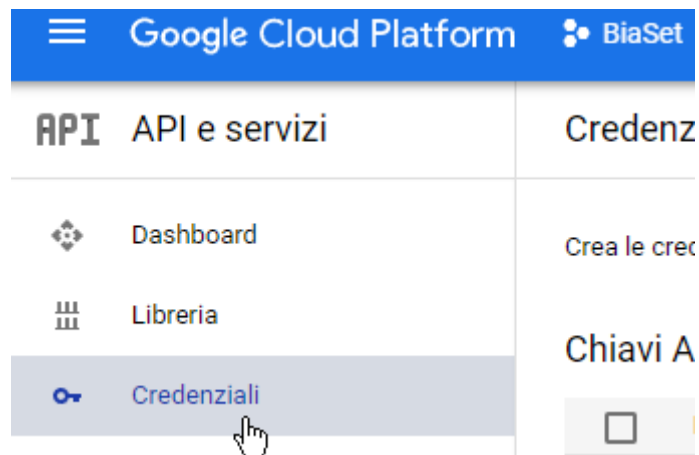


Figura 4 Menu Google Cloud Platform

ID client OAuth 2.0

<input type="checkbox"/>	Nome	Data di creazione ↓
<input type="checkbox"/>	authapp-biaset	14 ago 2021

Nome *

authapp-biaset

Il nome del client OAuth 2.0. È utilizzato solo per identificare il client nella console e non verrà visualizzato dagli utenti finali.

ID client

79975022959

upvh04.apps.googleusercontent.com

Client secret

Data di creazione

14 agosto 2021 18:22:24 GMT+2

Figura 5 Credenziali Google

- Per **Facebook**, invece, nelle **Impostazioni di base** del progetto creato, sotto la voce “ID App”.

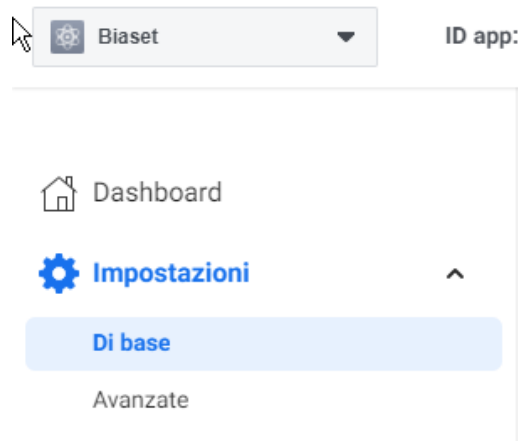


Figura 6 Menu Facebook for Developers

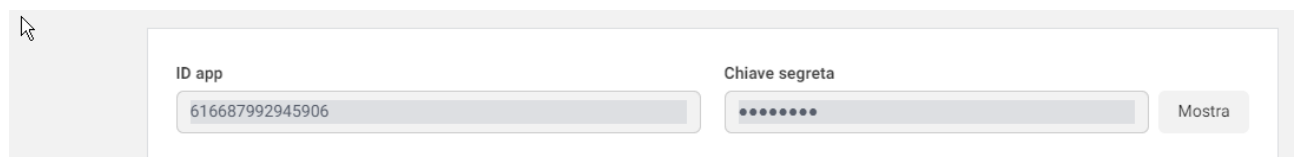


Figura 7 Credenziali Facebook

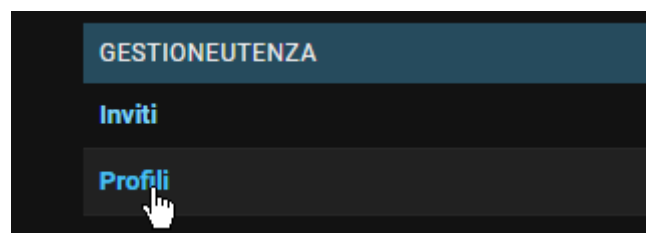
- **Secret Key:** inserire la chiave segreta delle due applicazioni. È identificabile nelle figure 5 e 7, rispettivamente come “Client secret” e “Chiave Segreta”
- **Key:** vuoto
- **Sites:** selezionare il sito “localhost”, doppio cliccando su di esso.
- **Cliccare Su SALVA.**

È necessario effettuare l’operazione per entrambi i Social.

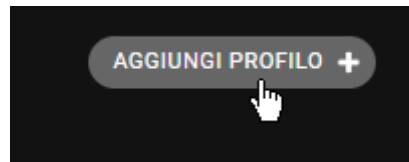
Ora la web app è correttamente configurata per interagire con Google e Facebook per quanto riguarda la registrazione e il login.

3.5 Configurazione Profili

Tornare sulla Home Page dell’amministrazione e cliccare su Profili.



Caricata la pagina, procedere alla creazione di **3 profili** attraverso il tasto “Aggiungi profilo” in alto a destra.



Nella casella **Tipo profilo**, indicare il nome del profilo da aggiungere. In questo caso, dovranno essere effettuati 3 inserimenti con valori differenti, ovvero:

- **League Admin**
- **Championship Admin**
- **Allenatore**

Attenzione: è necessario rispettare l’esatta nomenclatura indicata, altrimenti è possibile che si verifichino dei malfunzionamenti della piattaforma.

La configurazione di base è terminata. La piattaforma risulta pienamente utilizzabile dagli utenti.

3.6 Configurazione E-mail

La piattaforma prevede anche l’invio automatizzato di e-mail per quanto riguarda gli Inviti. Per configurare tale servizio, sarà necessario recarsi nella cartella delle impostazioni */settings/* e modificare il file *dev.py*, in quanto stiamo procedendo all’installazione dell’app in locale.

All’interno del file *dev.py*, sarà necessario aggiungere le seguenti stringhe:

```
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
EMAIL_USE_TLS = False
EMAIL_HOST = 'email-host'
EMAIL_HOST_USER = 'email-username'
EMAIL_HOST_PASSWORD = 'email-password'
EMAIL_PORT = server-port
```

Ovviamente, dovranno essere sostituiti i valori dei campi EMAIL_HOST, EMAIL_HOST_USER, EMAIL_HOST_PASSWORD ed EMAIL_PORT con i dati forniti dal vostro host del servizio mail.



4. Glossario

In questa sezione sono raccolti tutti i termini e le sigle che necessitano di una definizione:

- **Biaset:** nome della web application che si andrà ad installare;
- **Docker:** software utile per la virtualizzazione a livello di SO;
- **ORM:** Object Relational Mapper per interagire con i database;
- **Django:** framework utilizzato per lo sviluppo della web application;
- **Google Cloud Platform:** piattaforma con suite di servizi offerti da Google;
- **Facebook for Developers:** piattaforma con suite di servizi offerti da Facebook;
- **API:** application programming interface, ovvero librerie software per effettuare operazioni;
- **OAuth 2.0:** evoluzione del protocollo di rete OAuth 1.0. consente l'emissione di un token di accesso da parte di un server autorizzativo ad un client di terze parti, previa approvazione dell'utente proprietario della risorsa cui si intende accedere
- **Django Admin:** è la libreria che Django offre per una gestione di tutti i modelli del framework.