



CerCollettiva - Setup

Software opensource per la gestione delle comunità energetiche

09/06/2024

Andrea Bernardi

Venetiae Progetti STP S.r.l.

Via delle Macchine 9/F


30038, Spinea (Ve)

Distribuito secondo licenza MIT



Indice della Documentazione per il setup di base

1. Installazione del Sistema Operativo (Raspbian):	2
2. Configurazione Iniziale:	3
3. Installazione di Apache, MariaDB e PHP:	3
4. Configurazione di Apache:	3
5. Configurazione di MariaDB:	4
6. Installazione di Composer:	6
7. Installazione di Laravel:	6
8. Installazione di Mosquitto:	7
Configurazione MQTT (Mosquitto)	7
10. Configurazione di Laravel:	11



Obiettivo: Preparare il Raspberry Pi 4 Model B come ambiente di sviluppo e test per il software CerCollettiva, utilizzando la configurazione LAMP (Linux, Apache, MariaDB, PHP) e il broker MQTT Mosquitto.

Prerequisiti:

- Raspberry Pi 4 Model B
- Alimentatore USB-C (5V, 3A)
- Scheda microSD (minimo 8GB)
- Cavo HDMI
- Tastiera e mouse USB
- Connessione a Internet

Passaggi:

1. **Installazione del Sistema Operativo (Raspbian):**

- Scarica l'ultima versione di Raspberry Pi Imager dal sito ufficiale:
<https://www.raspberrypi.com/software/>.
- Inserisci la scheda microSD nel tuo computer.
- Apri Raspberry Pi Imager, seleziona l'immagine del sistema operativo Raspbian (Raspberry Pi OS) e la scheda microSD.
- Clicca su "WRITE" per avviare l'installazione.
- Una volta completata l'installazione, inserisci la scheda microSD nel Raspberry Pi e collegalo all'alimentazione.

2. Configurazione Iniziale:

- Al primo avvio, segui la procedura di configurazione guidata per impostare la lingua, la password, la connessione Wi-Fi, ecc.
- Apri un terminale e aggiorna il sistema operativo e i pacchetti installati:

Bash

```
# Aggiorna l'elenco dei pacchetti disponibili e installa gli aggiornamenti
```

```
sudo apt update
```

```
sudo apt upgrade -y
```

```
# Pulisci la cache dei pacchetti per liberare spazio
```

```
sudo apt clean
```

3. Installazione di Apache::

- Installa i pacchetti necessari:

Bash

```
# Installa Apache
```

```
sudo apt-get install apache2 -y
```

```
# Aggiungi il repository per PHP 8.2
```

```
sudo wget -qO /etc/apt/trusted.gpg.d/php.gpg
```

```
https://packages.sury.org/php/apt.gpg
```

```
echo "deb https://packages.sury.org/php/ $(lsb_release -sc)
main" | sudo tee /etc/apt/sources.list.d/php.list

# Aggiorna l'elenco dei pacchetti disponibili dopo
l'aggiunta del nuovo repository

sudo apt update

# Installa PHP 8.2 e i moduli necessari

sudo apt install -y php8.2 php8.2-common php8.2-cli
php8.2-mbstring php8.2-xml php8.2-pdo php8.2-mysql
php8.2-curl php8.2-zip unzip libapache2-mod-php8.2
```

- Verifica l'installazione di Apache aprendo un browser e navigando su `http://localhost`. Dovresti vedere la pagina di benvenuto di Apache.

```
# Rimuovi il file index.html di default di Apache

sudo rm /var/www/html/index.html

# Riavvia Apache per applicare le modifiche

sudo systemctl restart apache2
```

4. Configurazione di Apache:

- Crea un file di configurazione per il virtual host di CerCollettiva:

Bash

```
Sudo mkdir CerCollettiva/public

sudo nano /etc/apache2/sites-available/cercollettiva.conf
```

- Inserisci il seguente contenuto (sostituendo "tuo_dominio.it" con il tuo dominio, se applicabile):

```
<VirtualHost *:80>
    ServerName tuo_dominio.it
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/cercollettiva/public
    <Directory /var/www/html/cercollettiva/public>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

- Abilita il virtual host e riavvia Apache:

Bash

```
sudo a2ensite cercollettiva.conf

sudo a2dissite 000-default.conf

sudo a2enmod rewrite

sudo systemctl restart apache2
```

5. Installazione e Configurazione di MariaDB:

Bash

```
sudo apt install mariadb-server
```

Imposta la password di root di MariaDB:

Bash

```
sudo mysql_secure_installation
```

- Accedi a MariaDB come root:

Bash

```
sudo mysql -u root -p
```

- Crea un nuovo database per CerCollettiva (sostituisci "cercollettiva_database" con il nome desiderato):

SQL

```
CREATE DATABASE cercollettiva_database;
```

- Crea un nuovo utente per il database (sostituisci "nome_utente" e "password" con le credenziali desiderate):

SQL

```
CREATE USER 'cercollettiva'@'localhost' IDENTIFIED BY  
'*metti qua la tua password*';
```

- Concedi all'utente i permessi necessari sul database:

SQL

```
GRANT ALL PRIVILEGES ON cercollettiva_database.* TO  
'cercollettiva'@'localhost';
```

- Esci da MariaDB:

SQL

```
EXIT;
```

6. Installazione di Composer:

- Scarica e installa Composer:

Bash

```
curl -sS https://getcomposer.org/installer | php  
  
sudo mv composer.phar /usr/local/bin/composer
```

7. Installazione di Laravel:

- Installa Laravel globalmente:

Bash

```
composer global require laravel/installer  
  
export PATH="$PATH:$HOME/.config/composer/vendor/bin"
```


Crea un nuovo progetto Laravel nella directory `/var/www/html`:

Bash

```
cd /var/www/html

sudo chown -R $USER:$USER /var/www/html

laravel new cercollettiva
```

- Concedi i permessi necessari alla directory del progetto:

Bash

```
sudo chown -R www-data:www-data /var/www/html/cercollettiva

sudo chmod -R 755 /var/www/html/cercollettiva/storage
```

8. Configurazione di Laravel:

Apri il file `.env` nella directory principale del progetto Laravel. Questo file contiene le variabili d'ambiente che Laravel utilizza per la configurazione. Ecco le variabili più importanti da configurare:

- **APP_NAME**: Il nome della tua applicazione (es. `CerCollettiva`).
- **APP_ENV**: L'ambiente in cui stai eseguendo l'applicazione (`local`, `production`, ecc.).
- **APP_KEY**: Una chiave casuale utilizzata per la sicurezza dell'applicazione. Puoi generarla con il comando `php artisan key:generate`.

- **APP_DEBUG**: Imposta su `true` in fase di sviluppo per visualizzare gli errori dettagliati, `false` in produzione.
- **DB_CONNECTION**: Il tipo di database che stai utilizzando (`mysql` o `pgsql`).
- **DB_HOST**: L'host del database (es. `127.0.0.1` o `localhost`).
- **DB_PORT**: La porta del database (es. `3306` per MySQL, `5432` per PostgreSQL).
- **DB_DATABASE**: Il nome del database che hai creato per CerCollettiva.
- **DB_USERNAME**: Il nome utente del database.
- **DB_PASSWORD**: La password del database.
- **MAIL_MAILER**: Il driver per l'invio di email (es. `smtp`, `mailgun`, ecc.).
- **MAIL_HOST**: L'host del server SMTP.
- **MAIL_PORT**: La porta del server SMTP.
- **MAIL_USERNAME**: Il nome utente per l'autenticazione SMTP.
- **MAIL_PASSWORD**: La password per l'autenticazione SMTP.
- **MAIL_ENCRYPTION**: Il tipo di crittografia da utilizzare (`tls` o `null`).
- **MAIL_FROM_ADDRESS**: L'indirizzo email del mittente.
- **MAIL_FROM_NAME**: Il nome del mittente.

Esempio di file `.env`:

```
APP_NAME=CerCollettiva

APP_ENV=local

APP_KEY=base64:chiave_casuale_generata

APP_DEBUG=true

DB_CONNECTION=mysql
```

```
DB_HOST=127.0.0.1

DB_PORT=3306

DB_DATABASE=cercollettiva_database

DB_USERNAME=cercollettiva

DB_PASSWORD=*metti qua la tua password*

MAIL_MAILER=smtp

MAIL_HOST=smtp.example.com

MAIL_PORT=587

MAIL_USERNAME=null

MAIL_PASSWORD=null

MAIL_ENCRYPTION=null

MAIL_FROM_ADDRESS=hello@example.com


MAIL_FROM_NAME="${APP_NAME}"
```

Note Importanti:

9. Sostituisci i valori di esempio con le tue credenziali e impostazioni effettive.
10. Genera una nuova **APP_KEY** con il comando `php artisan key:generate`.
11. Assicurati che il database sia stato creato e che l'utente abbia i permessi corretti.
12. Configura le impostazioni email in base al tuo provider di posta elettronica.

```
php artisan migrate
```

Installazione di Node.js e NPM:



```
Installa Node.js e NPM:
```

```
Bash
```

```
sudo apt install nodejs npm
```

```
Installa le dipendenze frontend di Laravel:
```

```
Bash
```

```
cd /var/www/html/cercollettiva
```

```
Sudo npm install
```

```
Sudo npm run build
```

13. Installazione di Mosquitto:

- Installa Mosquitto:

```
Bash
```

```
sudo apt install mosquitto mosquitto-clients
```

- Configura Mosquitto per l'autenticazione e la gestione dei topic

Ecco una possibile configurazione MQTT:

Configurazione MQTT (Mosquitto)

```
# Configurazione di base
```

```
listener 1883 # Porta predefinita per MQTT

protocol mqtt

# Persistenza dei messaggi

persistence true

persistence_location /var/lib/mosquitto/ # Directory per i
dati persistenti

# Autenticazione

allow_anonymous false

password_file /etc/mosquitto/pwfile

# Logging

log_dest file /var/log/mosquitto/mosquitto.log

# Configurazione TLS (opzionale, ma consigliata)

listener 8883

cafile /etc/mosquitto/ca_certificates/ca.crt

certfile /etc/mosquitto/certs/server.crt

keyfile /etc/mosquitto/certs/server.key
```

Spiegazione:

- `listener 1883`: Imposta la porta predefinita per le connessioni MQTT senza TLS.
- `protocol mqtt`: Specifica il protocollo MQTT.
- `persistence true`: Abilita la persistenza dei messaggi, in modo che i messaggi non consegnati vengano salvati e consegnati successivamente.

- `persistence_location /var/lib/mosquitto/`: Imposta la directory in cui verranno salvati i dati persistenti.
- `allow_anonymous false`: Disabilita le connessioni anonime, richiedendo l'autenticazione per tutti i client.
- `password_file /etc/mosquitto/pwfile`: Specifica il file contenente le credenziali degli utenti (username e password). Questo file deve essere creato separatamente e le password devono essere criptate.
- `log_dest file /var/log/mosquitto/mosquitto.log`: Imposta il file di log per Mosquitto.

Configurazione TLS (Opzionale):

- `listener 8883`: Abilita una seconda porta (8883) per le connessioni MQTT con TLS.
- `cafile`, `certfile`, `keyfile`: Specificano i percorsi dei certificati CA, del certificato del server e della chiave privata del server, rispettivamente. Questi file sono necessari per abilitare la crittografia TLS.

Note Importanti:

- **File pwfile**: Il file `pwfile` deve essere creato nella directory `/etc/mosquitto/` e deve contenere una riga per ogni utente, nel formato `username:password_criptata`. Puoi utilizzare il comando `mosquitto_passwd` per creare e gestire gli utenti e le password.
- **Certificati TLS**: Se decidi di abilitare TLS, dovrai ottenere i certificati appropriati e configurarli correttamente.

- **Permessi:** Assicurati che il file di configurazione di Mosquitto (`mosquitto.conf`) e il file `pwfile` abbiano i permessi corretti per garantire la sicurezza.

Passaggi Aggiuntivi:

1. **Riavvia Mosquitto:** Dopo aver modificato il file di configurazione, riavvia Mosquitto per applicare le modifiche:

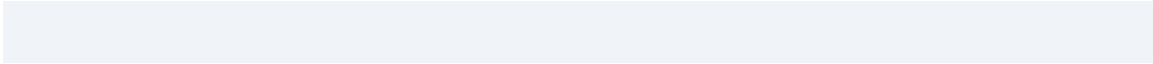

Bash

```
sudo systemctl restart mosquitto
```

2. **Verifica la Configurazione:** Utilizza un client MQTT (come `mosquitto_sub` e `mosquitto_pub`) per testare la connessione e l'autenticazione al broker.

Consigli:

- **Sicurezza:** La sicurezza è fondamentale quando si lavora con MQTT, soprattutto se si trasmettono dati sensibili. Assicurati di utilizzare password forti, crittografare la comunicazione con TLS e limitare l'accesso al broker solo ai dispositivi autorizzati.
- **Topic:** Organizza i topic in modo logico e coerente per facilitare la gestione dei dati. Ad esempio, potresti utilizzare una struttura gerarchica come
`/cercollettiva/{community_id}/{plant_id}/{measurement_type}`.
- **Monitoraggio:** Monitora regolarmente i log di Mosquitto per identificare eventuali errori o problemi di connessione.



Con questi passaggi, il tuo Raspberry Pi sarà configurato come ambiente di sviluppo per CerCollettiva. Potrai accedere all'applicazione tramite il browser e iniziare a sviluppare e testare le funzionalità del software.