

give me proper setup in i am using in grafan in add data sourece in postgresql i need to add log in postgresql history status set up prometheus also

ChatGPT said:

ChatGPT

To set up **Grafana** with a **PostgreSQL** data source and integrate it with logs, history, status, and **Prometheus**, follow these steps:

---

## 1. Set up PostgreSQL as a Data Source in Grafana

### 1. Install Grafana:

For Debian/Ubuntu:

bash

Copy code

```
sudo apt-get install -y grafana
sudo systemctl start grafana-server
sudo systemctl enable grafana-server
```

○

### 2. Access Grafana:

- Open `http://<server-ip>:3000` in your browser.
- Default credentials: `admin / admin`.

### 3. Add PostgreSQL Data Source:

- Go to **Configuration > Data Sources > Add data source**.
- Choose **PostgreSQL**.
- Fill in the fields:
  - **Host:** `localhost:5432` (or your PostgreSQL server address)
  - **Database:** Name of your database.
  - **User:** PostgreSQL username.
  - **Password:** Password for the PostgreSQL user.
  - **SSL Mode:** Choose based on your setup (e.g., `disable` for localhost).

### 4. Test & Save.

---

## 2. Add Logs and History in PostgreSQL

Create a Table for Logs:

sql

Copy code

```
CREATE TABLE logs (
```

```
    id SERIAL PRIMARY KEY,  
    timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
    status TEXT,  
    message TEXT  
);
```

1.

#### Insert Data (Optional):

sql

Copy code

```
INSERT INTO logs (status, message) VALUES ('INFO', 'Grafana PostgreSQL  
setup successful');
```

2.

#### 3. Query Logs in Grafana:

- Create a **new panel**.

Use the query:

sql

Copy code

```
SELECT  
    timestamp AS time,  
    status,  
    message  
FROM logs  
ORDER BY timestamp DESC;
```

○

#### 4. Visualize Logs:

- Choose the **Table** or **Graph** visualization.

---

## 3. Set Up Prometheus

#### 1. Install Prometheus:

Download and configure Prometheus:

bash

Copy code

```
wget
```

```
https://github.com/prometheus/prometheus/releases/download/v2.47.0/prometheus-2.47.0.linux-amd64.tar.gz
tar -xvzf prometheus-2.47.0.linux-amd64.tar.gz
cd prometheus-2.47.0.linux-amd64
./prometheus --config.file=prometheus.yml
```

○

## 2. Add PostgreSQL Exporter:

Install the **PostgreSQL Exporter**:

bash

Copy code

wget

```
https://github.com/prometheus-community/postgres_exporter/releases/download/v0.13.0/postgres_exporter-0.13.0.linux-amd64.tar.gz
tar -xvzf postgres_exporter-0.13.0.linux-amd64.tar.gz
./postgres_exporter --data-source-name="user=postgres
password=<password> host=localhost port=5432 dbname=<database>"
```

○

Add **PostgreSQL Exporter** to Prometheus configuration:

yaml

Copy code

```
scrape_configs:
  - job_name: 'postgres'
    static_configs:
      - targets: ['localhost:9187']
```

○

## 3. Add Prometheus as a Data Source in Grafana:

- Go to **Configuration > Data Sources > Add data source**.
- Choose **Prometheus**.
- URL: <http://localhost:9090>.

## 4. Create Panels:

Query example for PostgreSQL metrics:

promql

Copy code

```
pg_stat_database_connections{dbname="your_database_name"}
```

○

---

## 4. Combine PostgreSQL Logs and Prometheus Metrics

- Create dashboards with:
  - **Prometheus metrics** (e.g., CPU, memory, database connections).
  - **PostgreSQL queries** (e.g., logs, status, history).
- Use Grafana **Alerts** to notify about anomalies:
  - Create conditions based on Prometheus or PostgreSQL metrics.
  - Configure alert channels (e.g., email, Slack).

Let me know if you'd like detailed assistance with any specific step!

4o

**You said:**

all setup in live database i need to loki also in log not in localhost live ip this is postgresql ip 192.168.6.208:5432 all core set up in ubuntu

**ChatGPT said:**

ChatGPT

To set up **Grafana** with a live **PostgreSQL** database on IP **192.168.6.208:5432** and integrate it with **Loki** for logging, along with Prometheus for monitoring, follow the comprehensive steps below. This setup assumes an Ubuntu environment.

---

## 1. Install and Configure PostgreSQL Exporter

To monitor the live PostgreSQL database, use the **PostgreSQL Exporter**.

**Install PostgreSQL Exporter:**

bash

Copy code

wget

```
https://github.com/prometheus-community/postgres_exporter/releases/download/v0.13.0/postgres_exporter-0.13.0.linux-amd64.tar.gz
tar -xvzf postgres_exporter-0.13.0.linux-amd64.tar.gz
mv postgres_exporter /usr/local/bin/
```

1.

**Set Up Data Source for Exporter:** Create a PostgreSQL user with read-only permissions:

sql

Copy code

```
CREATE USER exporter WITH PASSWORD 'exporter_password';
GRANT CONNECT ON DATABASE your_database TO exporter;
GRANT SELECT ON ALL TABLES IN SCHEMA public TO exporter;
ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT SELECT ON TABLES TO exporter;
```

2.

**Run PostgreSQL Exporter:**

bash

Copy code

```
postgres_exporter
--data-source-name="postgresql://exporter:exporter_password@192.168.6.
208:5432/your_database" &
```

3. The exporter will run on port **9187** by default.

---

## 2. Install Prometheus

**Download and Install Prometheus:**

bash

Copy code

```
wget
https://github.com/prometheus/prometheus/releases/download/v2.47.0/prometheus-2.47.0.linux-amd64.tar.gz
tar -xvzf prometheus-2.47.0.linux-amd64.tar.gz
mv prometheus-2.47.0.linux-amd64 /usr/local/bin/prometheus
```

1.

**Configure Prometheus:** Edit the configuration file `prometheus.yml`:

yaml

Copy code

```
global:
  scrape_interval: 15s
```

```
scrape_configs:
```

```
- job_name: 'postgres'
  static_configs:
    - targets: ['192.168.6.208:9187']
```

2.

#### Start Prometheus:

bash

Copy code

```
./prometheus --config.file=prometheus.yml &
```

3. Prometheus will run on port **9090** by default.

---

## 3. Install Loki for Centralized Logging

#### Install Loki:

bash

Copy code

wget

```
https://github.com/grafana/loki/releases/download/v2.9.1/loki-linux-amd64.zip
```

```
unzip loki-linux-amd64.zip
```

```
chmod +x loki-linux-amd64
```

```
mv loki-linux-amd64 /usr/local/bin/loki
```

1.

**Configure Loki:** Create a configuration file `loki-config.yaml`:

yaml

Copy code

```
server:
```

```
  http_listen_port: 3100
```

```
ingester:
```

```
  wal:
```

```
    enabled: true
```

```
  chunk_idle_period: 5m
```

```
  max_chunk_age: 1h
```

```
  chunk_target_size: 1572864
```

```
schema_config:
  configs:
    - from: 2022-05-01
      store: boltdb-shipper
      object_store: filesystem
      schema: v11
      index:
        prefix: index_
        period: 24h

storage_config:
  boltdb_shipper:
    active_index_directory: /tmp/loki/boltdb-shipper-active
    cache_location: /tmp/loki/boltdb-shipper-cache
    shared_store: filesystem

  filesystem:
    directory: /tmp/loki/chunks

limits_config:
  retention_period: 30d

chunk_store_config:
  max_look_back_period: 0s
```

2.

#### Run Loki:

bash

Copy code

```
loki --config.file=loki-config.yaml &
```

3. Loki will run on port 3100.

---

## 4. Configure Grafana

#### Install Grafana:

bash

Copy code

```
sudo apt-get install grafana
sudo systemctl start grafana-server
sudo systemctl enable grafana-server
```

- 1.
2. **Access Grafana:**
  - Open: `http://<your-live-ip>:3000`.
  - Default credentials: `admin / admin`.
3. **Add Data Sources:**
  - **PostgreSQL:**
    - Navigate to **Configuration > Data Sources > Add data source**.
    - Select **PostgreSQL** and configure:
      - Host: `192.168.6.208:5432`
      - Database: `your_database`
      - User: `exporter`
      - Password: `exporter_password`
      - SSL: **Disable** (if no SSL is used).
    - Test and save.
  - **Prometheus:**
    - Navigate to **Configuration > Data Sources > Add data source**.
    - Select **Prometheus**.
    - URL: `http://<your-live-ip>:9090`.
    - Test and save.
  - **Loki:**
    - Navigate to **Configuration > Data Sources > Add data source**.
    - Select **Loki**.
    - URL: `http://<your-live-ip>:3100`.
    - Test and save.
4. **Create Dashboards:**
  - For PostgreSQL Logs:

Use queries like:

sql

Copy code

```
SELECT timestamp AS time, status, message
FROM logs
ORDER BY timestamp DESC;
```

■

- For Prometheus Metrics:



Query examples:

promql

Copy code

```
pg_stat_database_connections{datname="your_database"}
```

- - For Loki Logs:
      - Use `{job="your_job"}` to filter logs.
- 

## 5. Set Up Alerts in Grafana

### 1. Navigate to Alerts:

- Create alerts based on Prometheus or PostgreSQL metrics.

For example, alert when PostgreSQL connections exceed a threshold:

promql

Copy code

```
pg_stat_database_connections{datname="your_database"} > 50
```

- - 2. **Set Notification Channels:**
    - Configure email, Slack, or other integrations.
- 

## 6. Finalize Security

### 1. Secure Grafana:

- Enable HTTPS for Grafana.
- Configure strong passwords for all accounts.

### 2. Firewall Rules:

- Open only necessary ports (5432, 9090, 3100, 3000).

Use `ufw`:

bash

Copy code

```
sudo ufw allow 5432/tcp
```

```
sudo ufw allow 9090/tcp
```

```
sudo ufw allow 3100/tcp
```

```
sudo ufw allow 3000/tcp
```

-

Let me know if further details are needed!

4o

**You said:**

my live database in another pc then you give me proper setup

**ChatGPT said:**

ChatGPT

Here's the step-by-step guide for setting up **Grafana**, **Loki**, **Prometheus**, and connecting to a live PostgreSQL database hosted on another PC with IP **192.168.6.208:5432**. This guide ensures a complete setup for monitoring, logging, and visualization.

---

## 1. Prerequisites

- Ensure the database server (192.168.6.208) allows incoming connections:

Edit the PostgreSQL configuration file (**postgresql.conf**):

conf

Copy code

```
listen_addresses = '*'
```

○

Add the client IP of your Grafana/Prometheus server in **pg\_hba.conf**:

conf

Copy code

```
host all all <Grafana_Server_IP>/32 md5
```

○

Restart PostgreSQL:

bash

Copy code

```
sudo systemctl restart postgresql
```

○

Open the PostgreSQL port (**5432**) on the database server:

bash

Copy code

```
sudo ufw allow 5432/tcp
```

- 

---

## 2. Install and Configure PostgreSQL Exporter

To monitor the PostgreSQL database from Grafana via Prometheus:

**Download PostgreSQL Exporter:** On the server hosting Prometheus:

bash

Copy code

wget

```
https://github.com/prometheus-community/postgres_exporter/releases/download/v0.13.0/postgres_exporter-0.13.0.linux-amd64.tar.gz
tar -xvzf postgres_exporter-0.13.0.linux-amd64.tar.gz
sudo mv postgres_exporter /usr/local/bin/
```

1.

**Create a Read-Only User in PostgreSQL:** On the database server (192.168.6.208):

sql

Copy code

```
CREATE USER exporter WITH PASSWORD 'exporter_password';
GRANT CONNECT ON DATABASE your_database TO exporter;
GRANT SELECT ON ALL TABLES IN SCHEMA public TO exporter;
ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT SELECT ON TABLES TO exporter;
```

2.

**Run PostgreSQL Exporter:** On the Prometheus server:

bash

Copy code

```
postgres_exporter
--data-source-name="postgresql://exporter:exporter_password@192.168.6.208:5432/your_database" &
```

3. The exporter runs on port 9187 by default.

---

## 3. Install and Configure Prometheus

### Install Prometheus:

bash

Copy code

wget

```
https://github.com/prometheus/prometheus/releases/download/v2.47.0/prometheus-2.47.0.linux-amd64.tar.gz
tar -xvzf prometheus-2.47.0.linux-amd64.tar.gz
sudo mv prometheus-2.47.0.linux-amd64 /usr/local/bin/prometheus
```

1.

### Edit Prometheus Configuration: Create or edit `prometheus.yml`:

yaml

Copy code

```
global:
  scrape_interval: 15s

scrape_configs:
  - job_name: 'postgres'
    static_configs:
      - targets: ['192.168.6.208:9187']
```

2.

### Start Prometheus:

bash

Copy code

```
prometheus --config.file=prometheus.yml &
```

3. Prometheus runs on port 9090.

---

## 4. Install and Configure Loki

### Install Loki:

bash

Copy code

wget

```
https://github.com/grafana/loki/releases/download/v2.9.1/loki-linux-amd64.zip
unzip loki-linux-amd64.zip
chmod +x loki-linux-amd64
```

```
sudo mv loki-linux-amd64 /usr/local/bin/loki
```

1.

**Create Loki Configuration:** Create `loki-config.yaml`:

yaml

Copy code

```
server:
```

```
  http_listen_port: 3100
```

```
ingester:
```

```
  wal:
```

```
    enabled: true
```

```
  chunk_idle_period: 5m
```

```
  max_chunk_age: 1h
```

```
  chunk_target_size: 1572864
```

```
schema_config:
```

```
  configs:
```

```
    - from: 2022-05-01
```

```
      store: boltdb-shipper
```

```
      object_store: filesystem
```

```
      schema: v11
```

```
      index:
```

```
        prefix: index_
```

```
        period: 24h
```

```
storage_config:
```

```
  boltdb_shipper:
```

```
    active_index_directory: /var/lib/loki/boltdb-shipper-active
```

```
    cache_location: /var/lib/loki/boltdb-shipper-cache
```

```
    shared_store: filesystem
```

```
  filesystem:
```

```
    directory: /var/lib/loki/chunks
```

```
limits_config:
```

```
  retention_period: 30d
```

2.

### Run Loki:

bash

Copy code

```
loki --config.file=loki-config.yaml &
```

3. Loki runs on port 3100.
- 

## 5. Install and Configure Grafana

### Install Grafana:

bash

Copy code

```
sudo apt-get update
sudo apt-get install grafana
sudo systemctl start grafana-server
sudo systemctl enable grafana-server
```

- 1.
2. **Access Grafana:** Open [http://<Grafana\\_Server\\_IP>:3000](http://<Grafana_Server_IP>:3000) in your browser.  
Default credentials: [admin / admin](#).
3. **Add Data Sources:**
  - **PostgreSQL:**
    - Navigate to **Configuration > Data Sources > Add data source**.
    - Select **PostgreSQL**.
    - Fill in the details:
      - **Host:** [192.168.6.208:5432](#)
      - **Database:** [your\\_database](#)
      - **User:** [exporter](#)
      - **Password:** [exporter\\_password](#)
      - **SSL Mode:** [disable](#) (if no SSL).
    - Test and save.
  - **Prometheus:**
    - Navigate to **Configuration > Data Sources > Add data source**.
    - Select **Prometheus**.
    - URL: [http://<Prometheus\\_Server\\_IP>:9090](http://<Prometheus_Server_IP>:9090).
    - Test and save.
  - **Loki:**
    - Navigate to **Configuration > Data Sources > Add data source**.
    - Select **Loki**.
    - URL: [http://<Grafana\\_Server\\_IP>:3100](http://<Grafana_Server_IP>:3100).
    - Test and save.

#### 4. Create Dashboards:

PostgreSQL Logs:

sql

Copy code

```
SELECT timestamp AS time, status, message
FROM logs
ORDER BY timestamp DESC;
```

○

Prometheus Metrics:

promql

Copy code

```
pg_stat_database_connections{datname="your_database"}
```

○

---

## 6. Secure the Setup

**Firewall Rules:** On the database server:

bash

Copy code

```
sudo ufw allow from <Grafana_Server_IP> to any port 5432
```

On the Grafana server:

bash

Copy code

```
sudo ufw allow 3000/tcp
sudo ufw allow 3100/tcp
sudo ufw allow 9090/tcp
```

- 1.
2. **Enable HTTPS for Grafana:** Configure SSL certificates for secure access to Grafana.

---

With this setup, Grafana will monitor metrics from Prometheus, logs from Loki, and query live PostgreSQL data from the external database (192.168.6.208:5432). Let me know if further clarification is needed!

4o

**You said:**

You set by set in introction which setup in my pc live database means another pc

ChatGPT said:

ChatGPT

Here's a clear **step-by-step guide** to set up monitoring and logging for a **live PostgreSQL database** hosted on another PC (192.168.6.208) using **Grafana**, **Prometheus**, and **Loki**. This guide assumes you are setting up everything on your local PC (Ubuntu) while connecting to the live database on the other PC.

---

## Step 1: Verify Connectivity to the Live PostgreSQL Database

**Test Database Connectivity:** From your local PC, run:

bash

Copy code

```
psql -h 192.168.6.208 -U your_username -d your_database
```

1. Replace `your_username` and `your_database` with your PostgreSQL credentials.

If the connection fails, ensure:

- PostgreSQL on 192.168.6.208 is configured to accept remote connections.

`pg_hba.conf` on the database server includes your local PC's IP:

conf

Copy code

```
host all all <your_local_ip>/32 md5
```

- 
- PostgreSQL is listening on all interfaces (`listen_addresses = '*'` in `postgresql.conf`).

Port `5432` is open on the database server's firewall:

bash

Copy code

```
sudo ufw allow 5432/tcp
```

- 
- 

## Step 2: Install PostgreSQL Exporter on Your PC

PostgreSQL Exporter will gather metrics from the live database.



### Download and Install Exporter:

bash

Copy code

wget

```
https://github.com/prometheus-community/postgres_exporter/releases/download/v0.13.0/postgres_exporter-0.13.0.linux-amd64.tar.gz
tar -xvzf postgres_exporter-0.13.0.linux-amd64.tar.gz
sudo mv postgres_exporter /usr/local/bin/
```

1.

### Create a Monitoring User on the Database Server: On the database server (192.168.6.208):

sql

Copy code

```
CREATE USER exporter WITH PASSWORD 'exporter_password';
GRANT CONNECT ON DATABASE your_database TO exporter;
GRANT SELECT ON ALL TABLES IN SCHEMA public TO exporter;
ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT SELECT ON TABLES TO exporter;
```

2.

### Run PostgreSQL Exporter on Your PC:

bash

Copy code

postgres\_exporter

```
--data-source-name="postgresql://exporter:exporter_password@192.168.6.208:5432/your_database" &
```

3. The exporter runs on port 9187 by default.

---

## Step 3: Install and Configure Prometheus

### Download and Install Prometheus:

bash

Copy code

wget

```
https://github.com/prometheus/prometheus/releases/download/v2.47.0/prometheus-2.47.0.linux-amd64.tar.gz
tar -xvzf prometheus-2.47.0.linux-amd64.tar.gz
sudo mv prometheus-2.47.0.linux-amd64 /usr/local/bin/prometheus
```

1.

**Create Prometheus Configuration:** Edit `prometheus.yml` to include the PostgreSQL

Exporter:

yaml

Copy code

```
global:
  scrape_interval: 15s

scrape_configs:
  - job_name: 'postgres'
    static_configs:
      - targets: ['localhost:9187']
```

2.

**Start Prometheus:**

bash

Copy code

```
prometheus --config.file=prometheus.yml &
```

3. Prometheus runs on port 9090.

---

## Step 4: Install and Configure Loki

Loki will collect logs for visualization in Grafana.

**Download and Install Loki:**

bash

Copy code

```
wget
https://github.com/grafana/loki/releases/download/v2.9.1/loki-linux-amd64.zip
unzip loki-linux-amd64.zip
chmod +x loki-linux-amd64
sudo mv loki-linux-amd64 /usr/local/bin/loki
```

1.

**Create Loki Configuration:** Create `loki-config.yaml`:

yaml

Copy code

```
server:
  http_listen_port: 3100

ingester:
  wal:
    enabled: true
  chunk_idle_period: 5m
  max_chunk_age: 1h
  chunk_target_size: 1572864

schema_config:
  configs:
    - from: 2022-05-01
      store: boltdb-shipper
      object_store: filesystem
      schema: v11
      index:
        prefix: index_
        period: 24h

storage_config:
  boltdb_shipper:
    active_index_directory: /var/lib/loki/boltdb-shipper-active
    cache_location: /var/lib/loki/boltdb-shipper-cache
    shared_store: filesystem

  filesystem:
    directory: /var/lib/loki/chunks

limits_config:
  retention_period: 30d
```

2.

**Run Loki:**

bash

Copy code

```
loki --config.file=loki-config.yaml &
```

3. Loki runs on port 3100.
- 

## Step 5: Install and Configure Grafana

### Install Grafana:

bash

Copy code

```
sudo apt-get update
sudo apt-get install grafana
sudo systemctl start grafana-server
sudo systemctl enable grafana-server
```

- 1.
  2. **Access Grafana:** Open [http://<your\\_local\\_pc\\_ip>:3000](http://<your_local_pc_ip>:3000) in a browser.  
Default credentials: [admin](#) / [admin](#).
  3. **Add Data Sources:**
    - **PostgreSQL:**
      - Navigate to **Configuration > Data Sources > Add data source**.
      - Select **PostgreSQL** and configure:
        - Host: [192.168.6.208:5432](#)
        - Database: [your\\_database](#)
        - User: [exporter](#)
        - Password: [exporter\\_password](#)
        - SSL Mode: Disable (if no SSL is configured).
      - Test and save.
    - **Prometheus:**
      - Navigate to **Configuration > Data Sources > Add data source**.
      - Select **Prometheus**.
      - URL: <http://localhost:9090>.
      - Test and save.
    - **Loki:**
      - Navigate to **Configuration > Data Sources > Add data source**.
      - Select **Loki**.
      - URL: <http://localhost:3100>.
      - Test and save.
- 

## Step 6: Create Dashboards in Grafana

**PostgreSQL Metrics:** Use Prometheus metrics like:

`promql`

Copy code

```
pg_stat_database_connections{datname="your_database"}
```

1.

**Logs via Loki:** Query logs with:

plaintext

Copy code

```
{job="postgresql"}
```

2.

3. **Combine Metrics and Logs:** Create panels combining metrics from Prometheus and logs from Loki for better insights.

---

## Step 7: Secure the Setup

**Firewall Rules:** On your local PC:

bash

Copy code

```
sudo ufw allow 3000/tcp # Grafana
sudo ufw allow 3100/tcp # Loki
sudo ufw allow 9090/tcp # Prometheus
```

1.

2. **Enable HTTPS for Grafana:** Configure SSL for secure web access.

---

This setup integrates your live PostgreSQL database, logs, and metrics for monitoring and visualization in Grafana. Let me know if you need further clarification!

4o

**You said:**

i am getting in prometheus.yml file configuration problem then add live ip in database creating problem in site is not running this problem creating t

**ChatGPT said:**

ChatGPT

Let's troubleshoot your **Prometheus configuration problem** step by step and ensure your site connects to the live PostgreSQL database properly. I'll also clarify how to correctly set up the `prometheus.yml` file to monitor the PostgreSQL Exporter.

---

## 1. Common Prometheus Configuration Errors

Check for these typical issues in your `prometheus.yml` file:

### 1. YAML Formatting Issues:

- YAML is indentation-sensitive. Use **2 spaces** for indentation and avoid tabs.

Example of correctly formatted `prometheus.yml`:

yaml

Copy code

```
global:
  scrape_interval: 15s

scrape_configs:
  - job_name: 'postgres'
    static_configs:
      - targets: ['192.168.6.208:9187']
```

- 
- Ensure `targets` includes the **PostgreSQL Exporter IP and port**.

**Prometheus Service Not Restarted:** After editing `prometheus.yml`, restart Prometheus:

bash

Copy code

```
sudo systemctl restart prometheus
```

2.

**Prometheus Logs:** Check Prometheus logs for errors:

bash

Copy code

```
sudo journalctl -u prometheus -f
```

3. Look for misconfigurations or invalid syntax.

---

## 2. Fix PostgreSQL Exporter Connection

Ensure the PostgreSQL Exporter is running and reachable by Prometheus.

**Start PostgreSQL Exporter:** Run the exporter with the correct connection string:

bash

Copy code

```
postgres_exporter
--data-source-name="postgresql://exporter:exporter_password@192.168.6.208:5432/your_database" &
```

1. Replace `exporter`, `exporter_password`, and `your_database` with your credentials.

**Verify Exporter is Working:** Access the exporter metrics in a browser or via `curl`:

bash

Copy code

```
curl http://192.168.6.208:9187/metrics
```

2. You should see a list of metrics. If not:
  - Verify that the PostgreSQL credentials are correct.
  - Ensure the PostgreSQL database allows remote connections (`pg_hba.conf`).

---

### 3. Prometheus Not Running or Connecting

If Prometheus isn't running or cannot connect:

**Verify Prometheus Service:**

bash

Copy code

```
sudo systemctl status prometheus
```

- 1.

**Test Connection to Exporter:** From the Prometheus server, ensure the exporter is reachable:

bash

Copy code

```
curl http://192.168.6.208:9187/metrics
```

- 2.

**Edit `prometheus.yml`:** Ensure the IP and port of the PostgreSQL Exporter are correct:

yaml

Copy code

```
scrape_configs:
  - job_name: 'postgresql'
```

```
static_configs:
  - targets: ['192.168.6.208:9187'] # PostgreSQL Exporter IP and
port
```

3.

**Reload Prometheus:** After editing:

bash

Copy code

```
sudo systemctl reload prometheus
```

4.

---

## 4. Debugging the Website Issue

If your **Grafana or site isn't running**, ensure:

**Grafana Service is Active:**

bash

Copy code

```
sudo systemctl status grafana-server
```

If not, start it:

bash

Copy code

```
sudo systemctl start grafana-server
```

1.

**Access Grafana:** Open your browser and navigate to:

arduino

Copy code

```
http://<your_local_pc_ip>:3000
```

2. If it doesn't load:

Ensure your firewall allows port **3000**:

bash

Copy code

```
sudo ufw allow 3000/tcp
```

○



Verify Grafana logs for errors:

bash

Copy code

```
sudo journalctl -u grafana-server -f
```

○

---

## 5. Check PostgreSQL Server Settings

On the live PostgreSQL server (192.168.6.208):

### 1. Allow Remote Connections:

Edit `postgresql.conf`:

conf

Copy code

```
listen_addresses = '*'
```

○

Restart PostgreSQL:

bash

Copy code

```
sudo systemctl restart postgresql
```

○

**Update `pg_hba.conf`:** Add an entry for your local PC:

conf

Copy code

```
host all all <your_local_pc_ip>/32 md5
```

2. Replace `<your_local_pc_ip>` with your Ubuntu PC's IP address.

**Open PostgreSQL Port in Firewall:**

bash

Copy code

```
sudo ufw allow 5432/tcp
```

3.

**Verify Database Access:** From your Ubuntu PC:

bash

Copy code

```
psql -h 192.168.6.208 -U exporter -d your_database
```

4.

---

## 6. Check Logs for Detailed Errors

For each service, check logs if issues persist:

### Prometheus Logs:

bash

Copy code

```
sudo journalctl -u prometheus -f
```

- 

### Grafana Logs:

bash

Copy code

```
sudo journalctl -u grafana-server -f
```

- 

### PostgreSQL Logs: Check logs on the live PostgreSQL server:

bash

Copy code

```
sudo tail -f /var/log/postgresql/postgresql-<version>-main.log
```

- 

---

## Let Me Know What Issues Persist!

- Provide specific error messages or behaviors (e.g., Prometheus fails to start, Grafana cannot connect).
- Share logs or screenshots to help diagnose the problem further.