PyScada的相关资料

PyScada是具有 HTML5 HMI 的开源 SCADA 系统，使用 Django 框架构建。

SCADA是什么

SCADA（Supervisory Control and Data Acquisition，监控与数据采集系统）是一种用于实时监控和控制工业过程的自动化系统。它通过采集、处理、传输和显示现场设备的数据，实现对生产过程的远程监控和管理。SCADA 系统在我国各个行业，**尤其是电力系统中有广泛的应用**，其技术发展也相对成熟。

SCADA 系统的主要功能如下：

1. **数据采集**：SCADA 系统通过各种传感器、执行器和控制器等设备采集现场数据，包括温度、压力、流量、位置等参数。
2. **数据处理**：SCADA 系统对采集到的数据进行处理，如滤波、标定、报警等，以便于用户分析和决策。
3. **数据传输**：SCADA 系统将处理后的数据传输至上级系统或云服务平台，以便于远程监控和数据分析。
4. **远程控制**：SCADA 系统可以根据需要对现场设备进行远程控制，如启停、调节参数等，实现实时调整生产过程的目的。
5. **报警与故障诊断**：SCADA 系统能够实时监测现场设备的状态，发现异常情况时及时发出报警，并可根据历史数据进行故障诊断。
6. **信息展示**：SCADA 系统通过图形化界面展示现场设备和生产过程的实时数据，便于操作人员了解现场情况。

SCADA 系统在我国的应用场景广泛，如**电力系统、燃气行业、石油与天然气工业**等。随着信息化技术的发展，SCADA 系统也在不断升级和优化，如从第一代基于单片机的 SCADA 系统发展到第三代基于分布式计算机网络和关系数据库的系统。未来，SCADA 系统将在更多行业和领域发挥重要作用，助力我国工业生产的自动化、智能化发展。

基于工业互联网的 SCADA 系统的显著特征

1. **高度集成与互联**：基于工业互联网的 SCADA 系统可以实现各种设备和系统的无缝集成，打破信息孤岛，实现数据的高度共享。通过工业互联网技术，设备、传感器、控制系统、企业信息系统等都可以连接到 SCADA 系统，提高数据的实时性、准确性和完整性。
2. **边缘计算与云计算相结合**：基于工业互联网的 SCADA 系统采用边缘计算与云计算相结合的方式，实现数据的实时处理和分析。边缘计算可以将部分数据处理任务部署在设备端，减少数据传输延迟，提高系统的响应速度。同时，云计算可以为大规模数据处理和分析提供强大的计算能力，从而满足企业对数据挖掘和智能决策的需求。
3. **数据安全与可靠性**：基于工业互联网的 SCADA 系统重视数据安全和可靠性，采用加密、认证、访问控制等技术手段，确保数据在传输、存储和处理过程中的安全性。此外，系统还能实现故障预警和自动切换，提高系统的稳定性和可靠性。
4. **人工智能与大数据分析**：基于工业互联网的 SCADA 系统可以利用人工智能和大数据技术对海量数据进行实时分析，挖掘潜在的价值信息。这有助于企业实现生产过程的优化调度、故障预测和智能决策。
5. **易用性与可扩展性**：基于工业互联网的 SCADA 系统采用模块化设计，具有良好的可扩展性。企业可以根据需要灵活配置系统功能，如增加监控点、扩展数据处理能力等。同时，系统还具有友好的用户界面，便于操作人员快速上手。
6. **低功耗与绿色环保**：基于工业互联网的 SCADA 系统采用先进的节能技术，如传感器休眠、数据压缩等，降低系统的功耗。这有助于企业降低运营成本，同时也有利于环境保护。

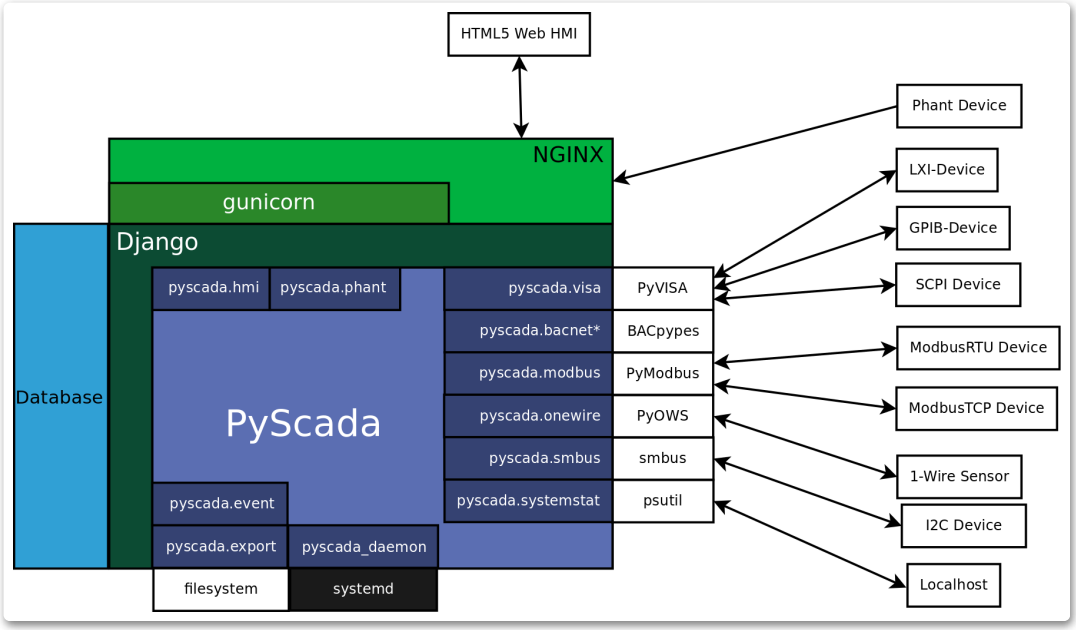
总之，基于工业互联网的 SCADA 系统在传统 SCADA 系统的基础上，融合了工业互联网的技术特点，具有更强大的数据处理、分析和管理能力，有助于企业实现生产过程的自动化、智能化和高效运行。

PyScada 的特点

- 基于 HTML5 的人机界面
- 支持以下内容
  - 工业协议
    - Modbus TCP/IP - RTU - ASCII - 二进制（使用pyModbus）

    - Phant（参见http://phant.io/）
    - VISA（使用pyVISA）
    - 1线
    - BACNet/IP（正在开发中）（使用BACpypes和BAC0）
    - MeterBus (MBus)（正在开发中）（使用pyMeterBus）
    - SMBus（使用smbus2）
    - GPIO（使用RPi.GPIO）
    - 系统统计
    - OPC-UA（使用opcua-asyncio）
    - SML（智能电表语言）（使用pySML）
    - 文件读/写
    - 串行
    - 网络服务
  - 设备
    - 通用虚拟设备
    - PT104（使用Pico PT-104）
  - 脚本
    - 脚本编写
  - 系统工具
  - 事件管理、数据导出、邮件通知
- 对服务器的硬件要求非常低

PyScada 的结构



PyScada 的安装

安装

本安装指南涵盖了在Debian 10/11和基于Raspberry Pi OS 的Linux 系统上安装 PyScada， 使用MariaDB作为数据库， 使用Gunicorn作为 WSGI HTTP 服务器，使用nginx作为 HTTP 服务器。

可用脚本

该脚本`install.sh`允许您在 2 种安装类型之间进行选择：系统或 docker，并创建安装日志文件。

然后它调用脚本`install_system.sh`或`install_docker.sh`根据您的选择。

在 Debian 及其衍生版本上自动安装

1. 选择下载 PyScada 的方法：

- 通过克隆存储库：

```
sudo apt install git
git clone https://github.com/pyscada/PyScada.git
cd PyScada
```

- 下载 zip 文件并解压：

```
sudo apt install wget
wget
https://github.com/pyscada/PyScada/archive/refs/heads/main
.zip -O PyScada-main.zip
sudo apt install unzip
unzip ./PyScada-main.zip
rm ./PyScada-main.zip
cd PyScada-main
```

2. 安装 PyScada

重要的：对于新安装，请务必对“仅更新”问题回答“否”。

您必须选择：

- 如果您想在系统上或 Docker 容器中安装 PyScada。
- 系统日期是否正确（仅限系统安装）
- 如果您想使用代理（仅限系统安装）
- 如果你想安装通道和redis来加速pyscada进程间的通信（仅限系统安装）
- 如果您只想更新，如果不想更新：
  - 数据库名称、用户和密码
  - 管理员名称和发送错误日志的邮件（需要在settings.py中进一步配置django电子邮件）
  - 第一个 pyscada 用户凭据
  - 如果您希望 pyscada 插件自动添加到 INSTALLED_APPS 中

run：

```
sudo ./install.sh
```

故障排除

如果您已经使用 docker 安装了 PyScada，则需要`db_data`使用以下命令删除 docker 卷：

```
docker volume rm docker_dbdata
```

PyScada 插件安装

1. 选择下载 PyScada 插件的方法（例如使用 PyScada-Modbus）：
   - 通过克隆存储库：

     ```
     sudo apt install git
     git clone https://github.com/pyscada/PyScada-Modbus.git
     cd PyScada-Modbus
     ```

   - 下载 zip 文件并解压：

```
sudo apt install wget
wget https://github.com/pyscada/PyScada-Modbus/archive/refs/heads/main.zip -O PyScada-Modbus-main.zip
sudo apt install unzip
unzip ./PyScada-Modbus-main.zip
rm ./PyScada-Modbus-main.zip
cd PyScada-Modbus-main
```

2. 安装 PyScada 插件

   run:

```
# activate the PyScada virtual environment
source /home/pyscada/.venv/bin/activate
# install the plugin
sudo -u pyscada -E env PATH=${PATH} pip3 install .
# run migrations
python /var/www/pyscada/PyScadaServer/manage.py migrate
# copy static files
python /var/www/pyscada/PyScadaServer/manage.py collectstatic --no-input
# restart gunicorn and PyScada
sudo systemctl restart gunicorn pyscada
```

列出已安装的 PyScada 插件

```
# activate the PyScada virtual environment
source /home/pyscada/.venv/bin/activate
pip3 list | grep cada
```

卸载插件

```
sudo -u pyscada -E env PATH=${PATH} pip uninstall yourPlugin
```

从旧版本更新

0.6.x 至 0.7.x

抱歉，无法直接升级，您必须从头开始安装 0.7.x。

0.7.0b18 至 0.7.0b19

```
cd /var/www/pyscada/PyScadaServer
sudo -u pyscada python manage.py migrate
sudo -u pyscada python manage.py collectstatic
sudo -u pyscada python manage.py pyscada_daemon init
```

系统

```
sudo wget
https://raw.githubusercontent.com/pyscada/PyScada/master/e
xtras/service/systemd/pyscada_daemon.service -O
/etc/systemd/system/pyscada_daemon.service
sudo systemctl enable pyscada_daemon
sudo systemctl disable pyscada_daq
sudo systemctl disable pyscada_event
sudo systemctl disable pyscada_mail
sudo systemctl disable pyscada_export
sudo rm /lib/systemd/system/pyscada_daq.service
sudo rm /lib/systemd/system/pyscada_mail.service
sudo rm /lib/systemd/system/pyscada_export.service
sudo rm /lib/systemd/system/pyscada_event.service
sudo systemctl daemon-reload
```

PyScada 的命令行

重新启动 PyScada 守护进程

systemd:

```
sudo systemctl restart pyscada
```

重新启动 Gunicorn

systemd:

```
sudo systemctl restart gunicorn.service
```

重新启动 NGINX

systemd:

```
sudo systemctl restart nginx
```

获取安装的 PyScada 版本

```
cd /var/www/pyscada/PyScadaServer
sudo -u pyscada python3 manage.py shell
import pyscada
pyscada.core.__version__
exit()
```

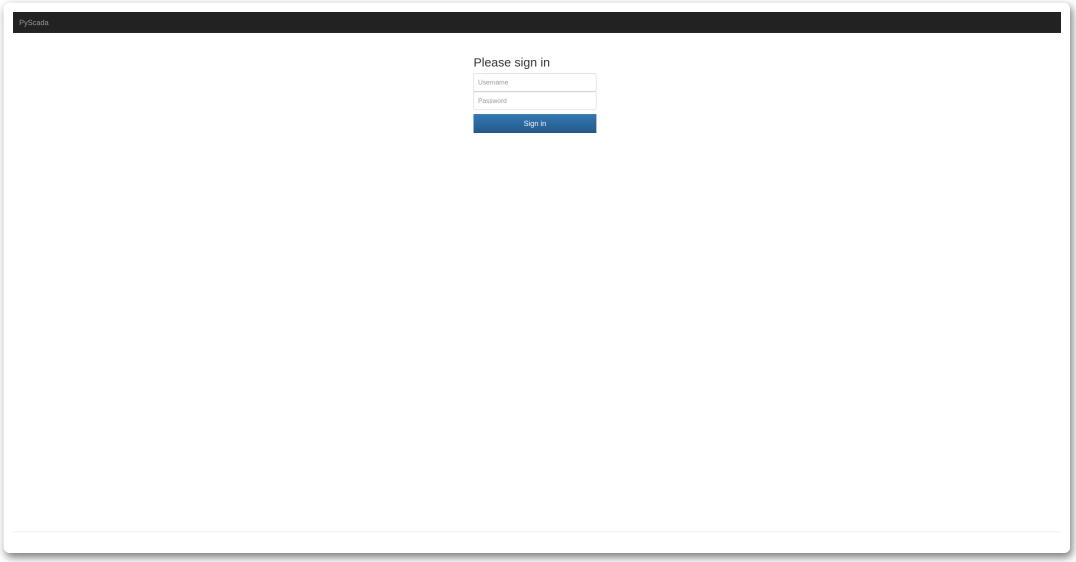
导出记录的数据表

```
sudo -u pyscada python3 manage.py PyScadaExportData # last
24 houres
sudo -u pyscada python3 manage.py PyScadaExportData --
start_time "01-03-2015 00:00:00" # from 01. of March 2015
until now
# from 01. of March until now, with the given filename
sudo -u pyscada python3 manage.py PyScadaExportData --
start_time "01-Mar-2015 00:00:00" --filename "filename.h5"
# from 01. of March until 10. of March, with the given
filename
sudo -u pyscada python3 manage.py PyScadaExportData --
start_time "01-03-2015 00:00:00" --filename "filename.h5"
--stop_time "10-03-2015 00:00:00"
```

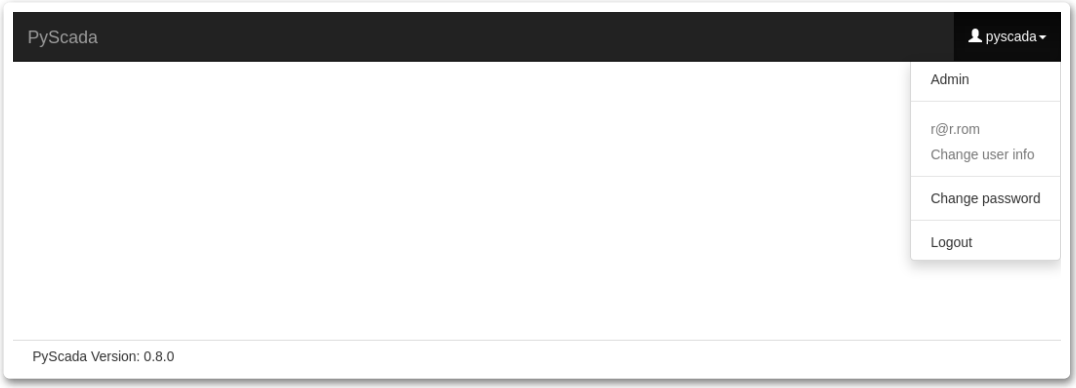
PyScada 的后端应用

使用后端

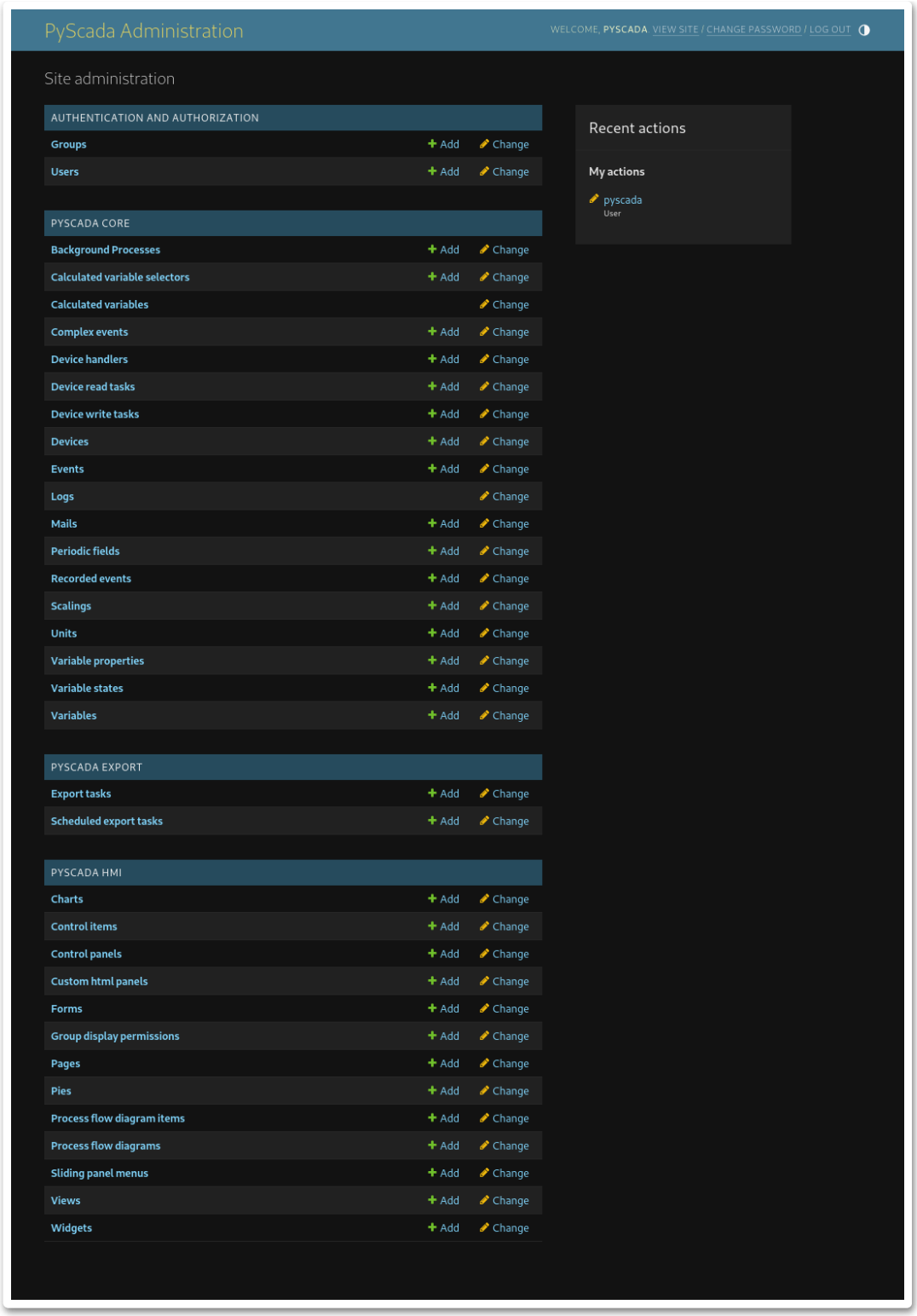
要使用后端，请在浏览器中打开http://127.0.0.1 （将 127.0.0.1 替换为 PyScada 服务器的 IP 或主机名），然后使用安装过程 中定义的管理员帐户登录 （TODO 链接到创建超级用户文档）。



成功登录后，您会看到视图概述，要打开管理面板，请单击右上角的用户名，然后单击Admin。

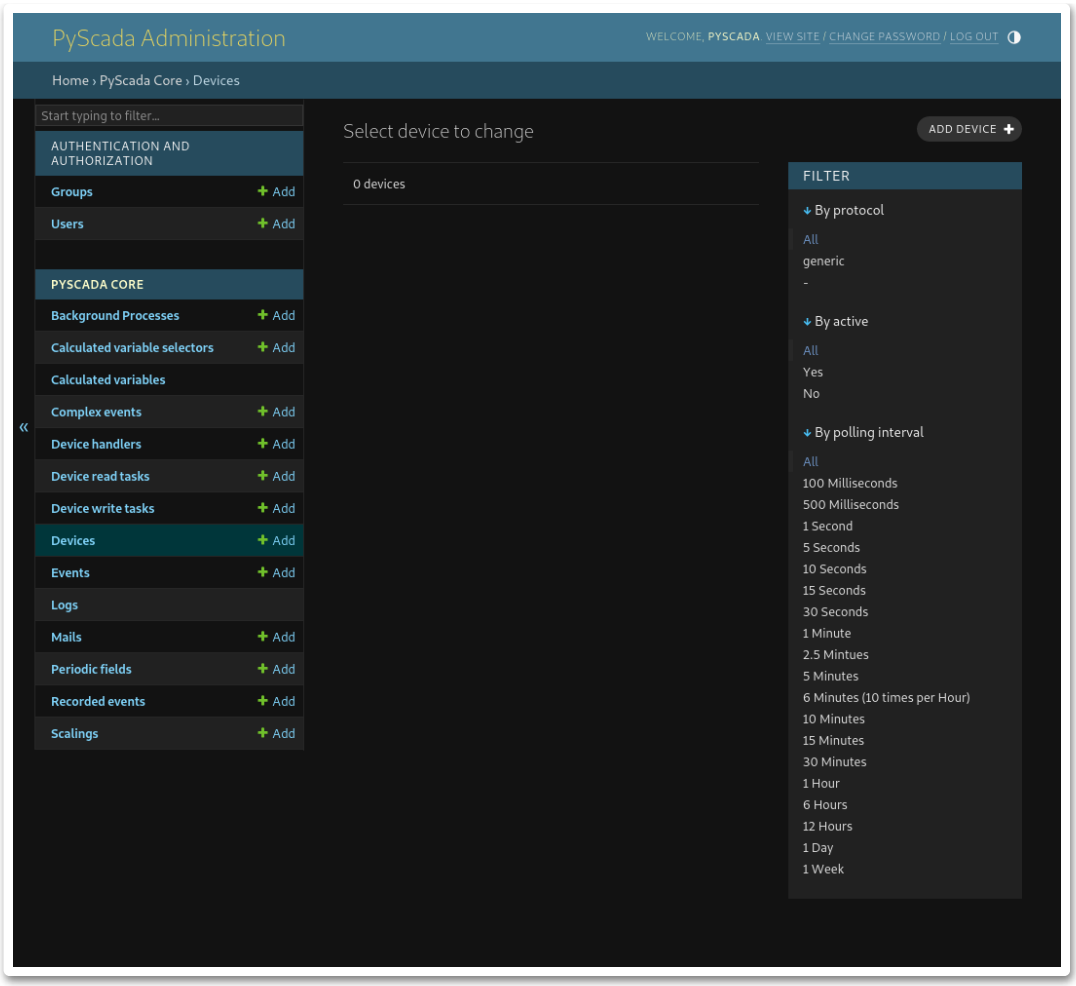


现在您位于后端或管理面板。

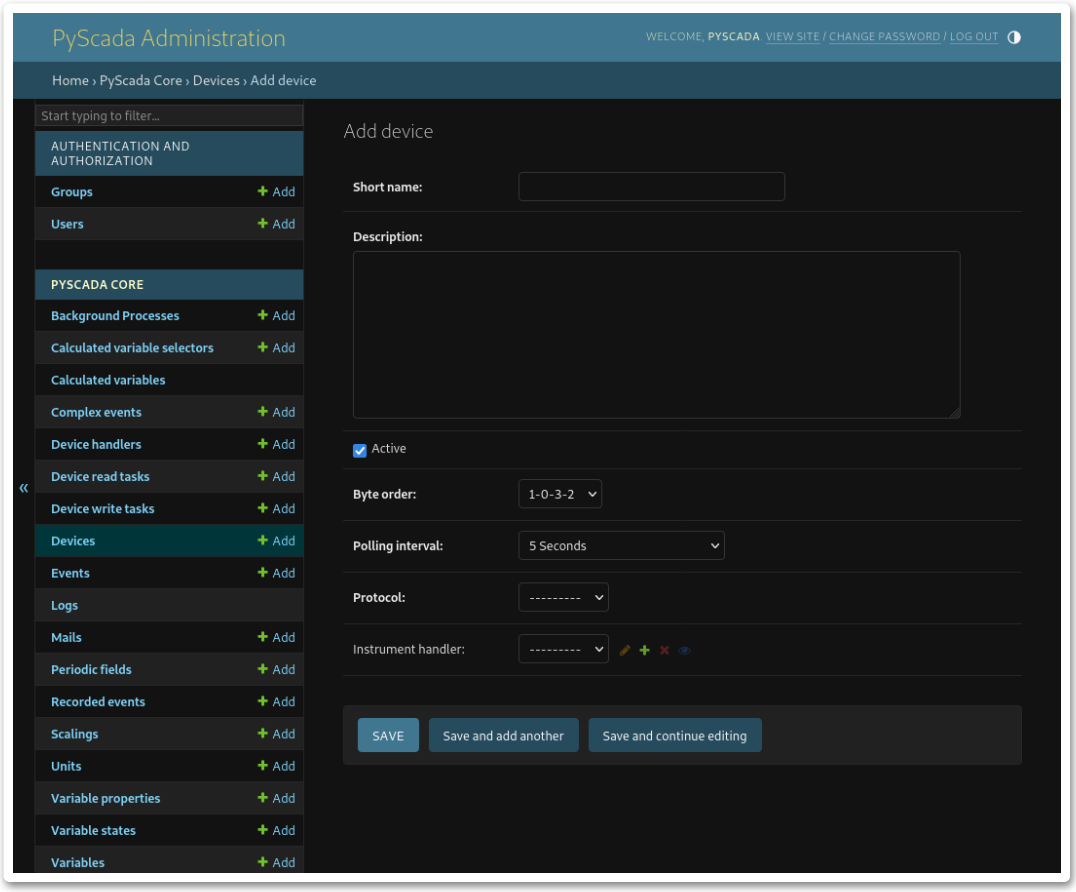


添加新设备

要添加新设备（例如 PLC），请打开PyScada Core部分中的设备表。

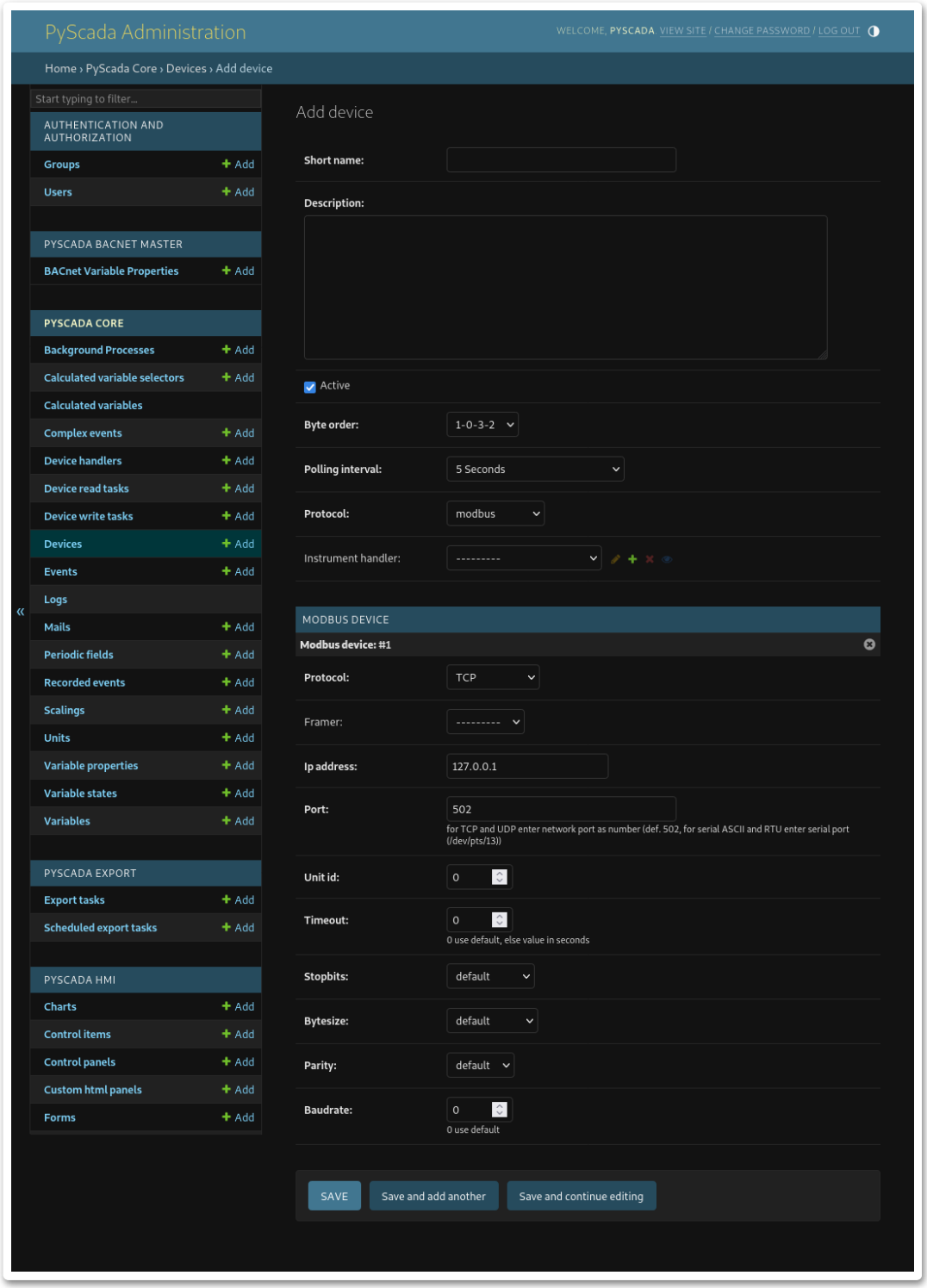


您将看到一个空列表。点击右上角的添加设备添加新设备（例如modbus设备）。



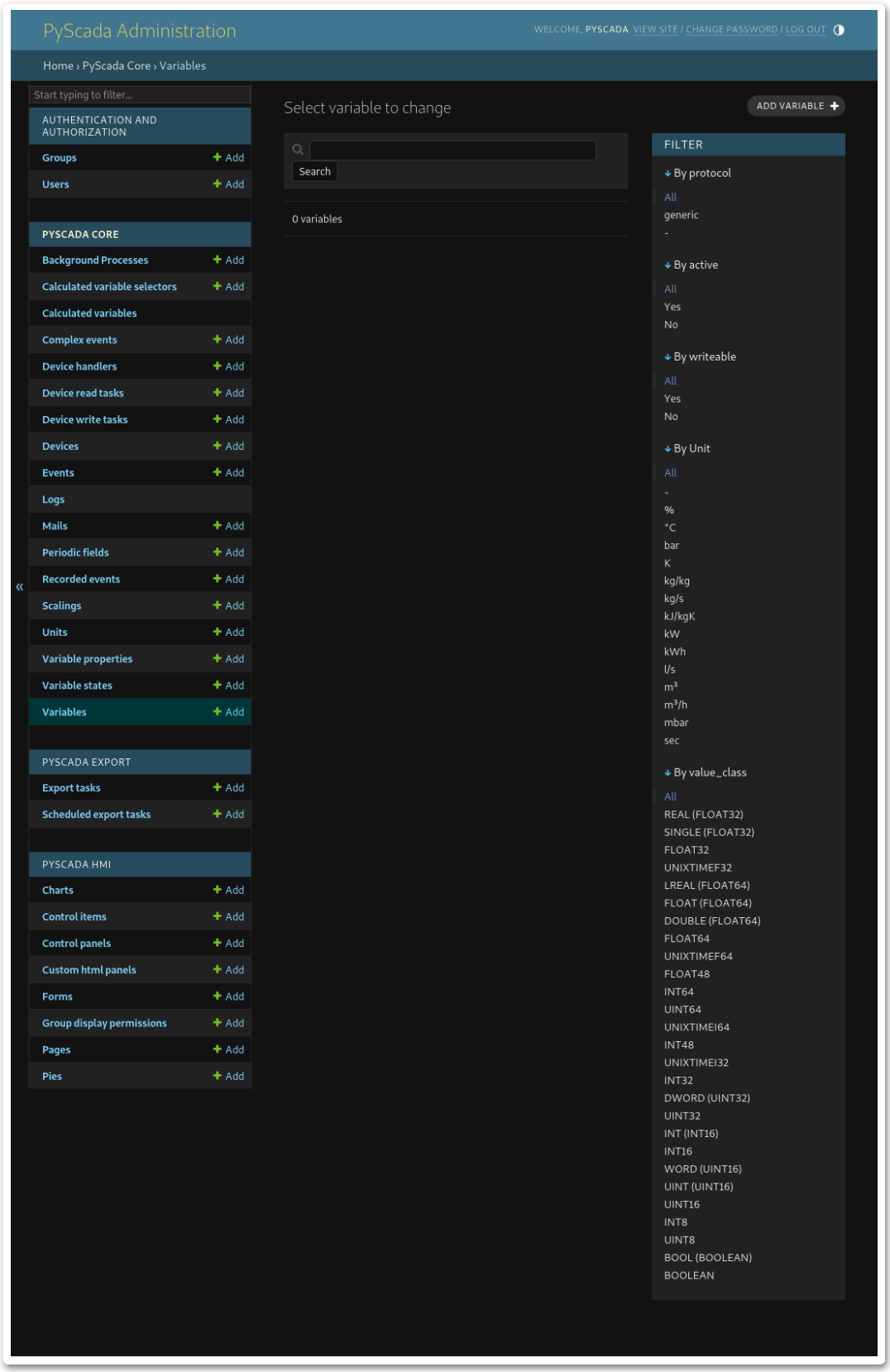
- 输入名称和描述。
- 选择池间隔（读取两个变量值之间的时间）。
- 选择一个协议。
- 输入该协议的必要信息。

(TODO设备协议设置)



添加新变量

进入管理面板PyScada Core部分的变量表。单击右上角的添加变量。



变量有名称和描述，将变量分配给设备并选择测量单位（TODO添加描述以添加新单位），如果应从 HMI 更改值，则激活可写，如果必须更改值缩放以便正确显示选择正确的缩放（TODO添加用于添加缩放的描述）。

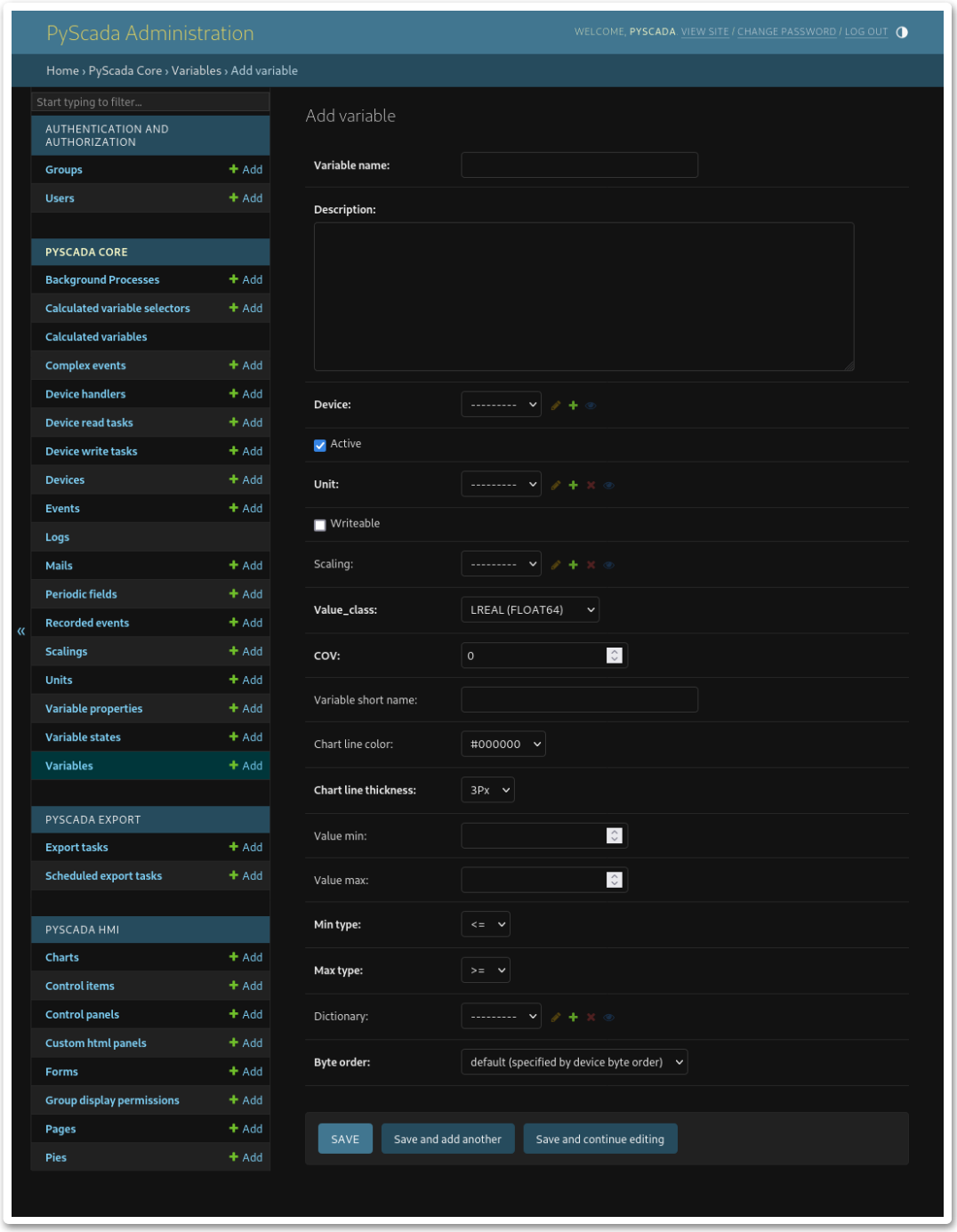
value_class是设备上表示值的数据类型（TODO添加示例）。

值变化(COV)是要存储在数据库中的值的变化量。如果满足以下条件，它将存储该变量的新值：(v=value)

$$|new_v - last_v| > COV_v$$

或者如果最后一个值早于 1 小时。

因此，如果您想保存所有值，请将 COV 设置为-1。



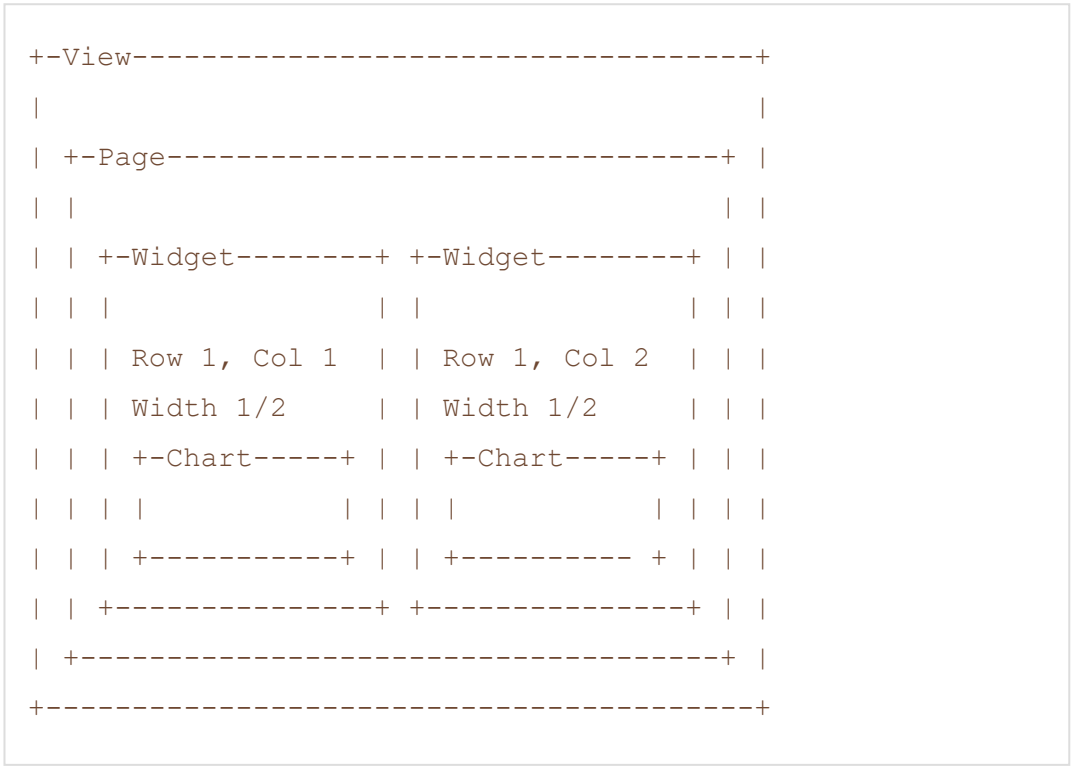
根据设备协议输入变量的必要信息。

构建用户 HMI（前端）的简短说明

在后端HMI部分：

1. 图表，添加新图表
2. 页面，添加页面
3. Widget，添加一个 Widget，在 Page 下选择您在 2 中添加的页面，并在Content下选择1 中的图表。
4. 小部件控制页面上每个元素的位置。设置小部件的位置（行、列）和宽度。
5. 查看，添加视图并从 2 中选择页面。
6. （可选）GroupDisplayPermissions，添加新的 GroupDisplayPermission，（如果需要添加新组并将您的用户添加到该组，请选择您在 1. 至 4 中创建的所有项目。）
7. 打开http://IP/，您应该看到新的视图，如果 DAQ 正在运行并且数据库中已有数据，您应该看到最近 2 小时的数据和当前数据。

前端结构：



设备协议 ID ¶

- 1：保留（调度程序）
- 2：系统统计
- 3：Modbus

- 4：BAC网络
- 5：签证
- 6：1线
- 7：潘特
- 8：SMBus
- 9：保留（Jofra350）
- 10：GPIO
- 11：保留（PT104）
- 12：OPC-UA
- 13：SML（智能电表语言）
- 14：文件
- 15：仪表总线（MBus）
- 16：通用虚拟设备
- 17：特快专递
- 18：运营
- 19：聚合
- 8X：定制工人
- 93：保留（串行）
- 94：保留（Web 服务）
- 95：保留（脚本）
- 96：保留（事件）
- 97：保留（邮件）
- 98：保留（报告）
- 99：保留（导出）
- 100+：保留用于动态

使用 Grafana

mysql

添加用户并授予 SELECT 权限：

```
sudo mysql -uroot -p -e "GRANT SELECT ON PyScada_db.* TO
'Grafana-user'@'localhost' IDENTIFIED BY 'Grafana-user-
password';"
```

nginx

在/etc/nginx/nginx.conf后添加：`http { ... }`

```
include /etc/nginx/grafana-access.conf;
```

使用以下命令创建/etc/nginx/grafana-access.conf：

```
stream {
  # MySQL server
  server {
    listen     3305;
    proxy_pass 127.0.0.1:3306;
    proxy_timeout 60s;
    proxy_connect_timeout 30s;
  }
}
```

重新启动 Nginx：

```
sudo systemctl restart nginx
```

Grafana

添加MySQL数据源：

- 主持人：
  - 本地：/run/mysqld/mysqld.sock
  - 远程：SERVER_WITH_NGINX_IP:3305
- 数据库：`PyScada_db`
- 用户：`Grafana-user`
- 密码：`Grafana-user-password`

创建仪表板：

- 或者导入示例仪表板。

- 或者例如，添加这些变量：set和：`refresh on dashboard load``multi-value``all option`
  - 添加 mysql 数据源变量（类型 Datasource）。
  - 使用类型查询添加变量`$Datasource`：
    - 协议：`SELECT protocol AS __text, id AS __value FROM pyscada_deviceprotocol`
    - 设备：`SELECT d.short_name AS __text, d.id AS __value FROM pyscada_device d WHERE d.protocol_id IN (${Protocols}) AND d.active = 1`
    - 单位：`SELECT u.unit AS __text, u.id AS __value FROM pyscada_unit u`
    - 变量：`SELECT v.name AS __text, v.id AS __value FROM pyscada_variable v WHERE v.device_id IN (${Devices}) AND v.unit_id IN (${Units}) AND v.active = 1`
    - 时间组（类型Interval）：`1s,10s,1m,10m,30m,1h,6h,12h,1d,7d,14d,30d,1M`
    - 空为（自定义类型）：`0, NULL, previous`

- 查询示例：

```
SELECT
  $__timeGroupAlias(r.date_saved,$time_group),
  avg(IFNULL(r.value_float64, 0.0) + IFNULL(r.value_int64, 0.0) + IFNULL(r.value_int32, 0.0) + IFNULL(r.value_int16, 0.0) + IFNULL(r.value_boolean, 0.0)),
  v.name AS metric
FROM pyscada_recordeddata as r
JOIN pyscada_variable v ON r.variable_id = v.id
WHERE
  $__timeFilter(r.date_saved) AND
  r.variable_id IN (${Variables})
GROUP BY time, metric
ORDER BY $__timeGroup(r.date_saved,$time_group,$null_as)
```

嵌入 pyscada HMI

编辑 Grafana 配置文件：

```
sudo nano /etc/grafana/grafana.ini
```

查找并设置：

- 允许嵌入= true
- 对于本地主机 grafana： root_url = http://localhost:3000/grafana/

对于 localhost grafana 添加/etc/nginx/sites-enabled/pyscada.conf：

```
location /grafana/ {
  proxy_pass http://127.0.0.1:3000/;
}
```

重新启动 Grafana 服务器：

```
sudo systemctl restart grafana-server.service
```

使用仪表板中的代码或 grafana 中共享选项的面板创建自定义 html 面板

其他

使用ssl：http://www.turbogeek.co.uk/2020/09/30/grafana-how-to-configure-ssl-https-in-grafana/

对于开发人员

激活PyScada虚拟环境

```
source /home/pyscada/.venv/bin/activate
```

克隆存储库

```
git clone git@github.com:pyscada/PyScada.git
```

对于像 PyScada-Modbus 这样的插件：

```
git clone git@github.com:pyscada/PyScada-Modbus.git
```

Pip 可编辑安装

激活虚拟环境后：

```
sudo -u pyscada -E env PATH=${PATH} pip3 install -e
./PyScada
```

对于像 PyScada-Modbus 这样的插件：

```
sudo -u pyscada -E env PATH=${PATH} pip3 install -e
./PyScada-Modbus
```

重新启动应用程序

激活虚拟环境后，要应用更改（根据更改的情况），可能需要：

创建迁移

```
python3 /var/www/pyscada/PyScadaServer/manage.py
makemigrations
```

应用它们

```
python3 /var/www/pyscada/PyScadaServer/manage.py migrate
```

复制静态文件（answer yes）

```
sudo -u pyscada -E env PATH=${PATH} python3
/var/www/pyscada/PyScadaServer/manage.py collectstatic
```

然后你可以：

对于 URL、视图或管理员更改，请重新启动 Gunicorn

```
sudo systemctl restart gunicorn
```

否则重新启动 PyScada

```
sudo systemctl restart pyscada
```

覆盖路线

当您希望重写现有视图（以及现有路线）时会遇到此用例。

PyScada 项目的`urls.py`文件用于加载软件的路由（请参阅此处）。

- python虚拟环境安装：位于`/var/www/pyscada/PyScadaServer/PyScadaServer`
- Docker 安装：位于`/src/pyscada/PyScadaServer/PyScadaServer`

默认情况下，项目的`urls.py`文件仅加载`urls.py`来自`pyscada.core`. 该文件以随机顺序`pyscada.core.urls`加载所有其他模块文件。`urls.py`

使用的路由是遇到的第一个有效路由，因此如果您想替换现有路由，则必须在其他路由之前加载您的路由，即在加载`pyscada.core.urls`文件之前。

为此，您需要修改项目的`urls.py`文件。

对于非 Docker 安装：

```
sudo -u pyscada nano
/var/www/pyscada/PyScadaServer/PyScadaServer/urls.py
```

并在 pyscada.core.urls 之前包含您的路线

```
urlpatterns = [
path('', include('pyscada.yourPlugin.urls')), #Routing
file yourPlugin
path('', include('pyscada.core.urls')),
]
```