
TPH forecast with Raspberry Pi and TensorFlow.

Mitsuhisa Tanaka

Nov 22, 2021

CONTENTS:

1	about	3
1.1	TPH forecast with Raspberry Pi and AI.	3
1.1.1	Preparing	3
1.1.2	Set up Raspberry Pi	4
1.1.3	Prepare development environment	5
1.1.4	Documentation	6
1.2	Development	6
1.2.1	Getting started	6
1.2.2	Requirements	6
1.2.3	make	9
1.2.4	Documentation	11
2	Indices and tables	13

Welcome to TPH forecast with Raspberry Pi and AI's documentation!

ABOUT

This project aim at forecasting temperature, pressure and humidity(TPH). And controlling air-conditioner. Home-automation with AI on Raspberry Pi and RPi TPH Monitor.

This is integrated your home automation, controlling air-conditioner with AI and another infrared controlled devices such as television set or ceiling light and so on.

1.1 TPH forecast with Raspberry Pi and AI.

I explain first boot your Raspberry Pi and set up Python environment for developing.

This project is developing now, yet not finish.

1.1.1 Preparing

Please get somethings next list.

- [Raspberry Pi 3B, 3B+](#)
- [RPi TPH Monitor Rev2](#)
- micro SD card, 16GB above(recommended)
- USB connected key board
- USB connected mouse
- [Raspbian](#)
- HDMI cable and display
 - use TV instead of display
- Python development environment
 - We supported only Python 3.7 upper version.

1.1.2 Set up Raspberry Pi

You must set up your Raspberry Pi.

On your Mac or PC(Linux, MS-Windows), you can install Raspbian to microSD card.

Download newest Raspbian

I recommend using official Raspbian which can download from [Raspberry Pi Downloads](#).

You will choose “Raspbian Buster with desktop and recommended software” or “Raspbian Buster with desktop”.

Installing operating system image

You must read [installation guide](#) for installing operating system image.

And download [balenaEtcher](#).

macOS

If you use Apple Mac, you can install via brew.

```
$ brew cask install balenaetcher
```

First boot

Only first boot time, You must connect USB keyboard, USB mouse, and monitor via HDMI. You must set Wi-Fi network and enable SSH via `raspbian-config`. Please set fixed IP address, for example `192.168.0.121/24`.

Test remote connect

On your Mac or PC, remote connecting test via ssh.

```
$ ssh pi@192.168.0.121
```

Package upgrade

I recommend upgrade your Raspbian.

```
$ sudo apt update
...
```

```
$ sudo apt upgrade
```


1.1.3 Prepare development environment

You can development on your Raspberry Pi.

I recommend preparing development environment on your Mac or PC.

pyenv and pyenv-virtualenv

Please install

macOS, Linux

- [pyenv](#)
- [pyenv-virtualenv](#)

Install Python via [PyEnv](#)

```
$ pyenv install 3.8.0
```

And setup pyenv-virtualenv

```
$ pyenv virtualenv 3.8.0 djrpi380
```

cf. my home directory.

```
$ pyenv versions
* system (set by /Users/mitsu/.pyenv/version)
 3.7.4
 3.7.4/envs/djsample374
 3.8.0
 3.8.0/envs/djrpi380
 djrpi380
 djsample374
```

```
$ python --version
Python 2.7.16
```

my environment directory.

```
$ cd ~/git/hub/django-rpi-tph-monitor
```

```
$ pyenv local djrpi380
```

```
$ pyenv versions
system
 3.7.4
 3.7.4/envs/djsample374
 3.8.0
 3.8.0/envs/djrpi380
* djrpi380 (set by /Users/mitsu/git/hub/django-rpi-tph-monitor/.python-version)
 djsample374
```

```
$ python --version  
Python 3.8.0
```

MS-Windows

If you use MS-Windows, [venv](#) instead of [pyenv](#).

Hydrogen

[Hydrogen](#) is an interactive coding environment that supports Python, R, JavaScript and other Jupyter kernels.

```
$ python -m ipykernel install --user --name=dj3rpi380 --display-name=dj3rpi380
```

If you multiple [pyenv](#) versions, you may repeat this command.

List your ipkernel.

```
$ jupyter kernelspec list
```

Let's begin development "Home automation application".

1.1.4 Documentation

We published documentation on [Read the Docs](#).

TPH forecast with Raspberry Pi and AI [documentation](#).

1.2 Development

This is setup protocol for developing Home automation with Raspberry PI and AI.

Not yet complete this project.

1.2.1 Getting started

1.2.2 Requirements

Python modules

- Django
- djangoestframework
- django-filter
- django-background-tasks
- dash
- plotly
- and so on.

```
$ cd tph
```

You may install Python packages via *pip*.

```
$ pip install -r requirements.txt
```

On your development Mac, Ubuntu, or MS-Windows.

```
$ pip install -r requirements_dev.txt
```

On your target Raspberry Pi.

About my Raspberry Pi 3B.

```
$ cat /etc/debian_version
10.9
$ uname -a
Linux raspib3 5.10.17-v7+ #1414 SMP Fri Apr 30 13:18:35 BST 2021 armv7l GNU/Linux
$ lsb_release -a
No LSB modules are available.
Distributor ID: Raspbian
Description:    Raspbian GNU/Linux 10 (buster)
Release:       10
Codename:      buster
$ cat /proc/device-tree/model
Raspberry Pi 3 Model B Rev 1.2
```

Enable i2c via raspi-config.

```
$ sudo raspi-config
```

Add i2c group your user account.

```
$ sudo usermod -aG i2c pi
```

Install MariaDB.

- [How to Install MariaDB on Raspberry Pi?](#)
- [Install MariaDB on Raspberry Pi OS](#)

```
$ sudo apt install mariadb-server
$ sudo mysql_secure_installation
Change the root password? [Y/n] y
Remove anonymous users? [Y/n] y
Disallow root login remotely? [Y/n] y
Remove test database and access to it? [Y/n] y
```

(continues on next page)

(continued from previous page)

```
Reload privilege tables now? [Y/n]
Cleaning up...
```

Install Python 'mysqlclient' module.

```
$ sudo apt install python3-dev default-libmysqlclient-dev build-essential
$ pip install mysqlclient
```

Setup timezone to MariaDB.

```
$ /usr/bin/mysql_tzinfo_to_sql /usr/share/zoneinfo > timezone.sql
$ mysql -u root -p -Dmysql < ./timezone.sql
```

Restart MariaDB.

```
$ sudo /etc/init.d/mysql restart
```

Install Python modules.

You should install another python modules.

```
$ pip install -r requirements_rpi.txt
```

And edit your tph/tph/settings.py

```
ON_RASPBERRY_PI = True
USE_SMBUS2 = True
```

recommended IDE(Integrated Development Environment)

- Atom ; base editor
- atom-ide ; make IDE base package
- ide-python ; support Atom-IDE Python language
- atom-ide-debugger-python ; DEBUG Python
- Hydrogen ; interactive coding environment in atom

setup for Hydrogen

```
$ pip install jupyter
$ python -m ipykernel install --user --name=<name> --display-name=<name>
$ jupyter kernelspec list
```

1.2.3 make

```
$ Python manage.py startapp monitor
```

```
$ Python manage.py makemigrations monitor
```

Set up your data base

```
$ Python manage.py migrate
```

```
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying sessions.0001_initial... OK
```

Using SCSS/SASS

Set up use [Sass](#) my Django project.

See and install [How to use SCSS/SASS in your Django Project\(Python Way\)](#).

Download Bootstrap Source file [here](#).

And copy SCSS files to static/bootstrap.

TPH forecast with Raspberry Pi and TensorFlow.

```
$ cp -r your/bootstrap-4.x.x/scss/* tph/static/bootstrap
```

Install some Python modules.

```
pip install django_compressor
pip install django_libsass
```

Background tasks

I selected [Django Background Tasks](#) for save datas interval.

For Django 3.2, `pip install django-background-tasks`.

```
pip install django-background-tasks
```

Registration background tasks and execute

First step

Create your Django Project.

```
mkdir django-rpi-tph-monitor
cd django-rpi-tph-monitor
```

```
django-admin startproject tph
cd tph
```

```
python manage.py runserver
```

django



The install worked successfully! Congratulations!

You are seeing this page because `DEBUG=True` is in your settings file and you have not configured any URLs.

Access `http://localhost:8000/` on your browser.

For access from remote computer to Raspberry Pi, on your Raspberry Pi:

```
python manage.py runserver 192.168.xxx.xxx:8000
```

You have to get another shell(terminal). Second registration task.

```
$ curl -X GET http://localhost:8000/monitor/tasks/5/30
```

Third run process tasks.

```
$ ./manage.py process_tasks
```

You can check tasks from your database that default is db.sqlite3. See background_task, background_task_completed_tasks, or monitor_bme280 tables.

1.2.4 Documentation

This project's documents are making with [SPHINX](#). How to use, please see [Installing Sphinx](#).

Note: If you are using PyEnv, you must install via pip.

```
$ pyenv virtualenv 3.9.4 dj32rpi394docs
$ cd ${your} django-rpi-monitor}/docs
$ pyenv local dj32rpi394docs
$ pip install --upgrade pip
$ pip install -r requirements.txt
```

Additional packages.

Sphinx-copybutton

[Sphinx-copybutton](#)

```
$ pip install --upgrade sphinx-copybutton
```

Read the Docs Theme

```
$ pip install --upgrade sphinx-rtd-theme
```

Making our documents.

You can create document.

```
cd docs
make html
```


INDICES AND TABLES

- `genindex`
- `modindex`
- `search`