PyCharm setup

This is a super-useful software that allows you to work on a local copy of your project and automatically deploy it to the workstations. To the best of our knowledge, it is the only IDE that supports **remote debugging** (the codes run on a remote server, but you can see the variables on your laptop). Read the guide!
To do so, you need a premium account by registering as academic here.

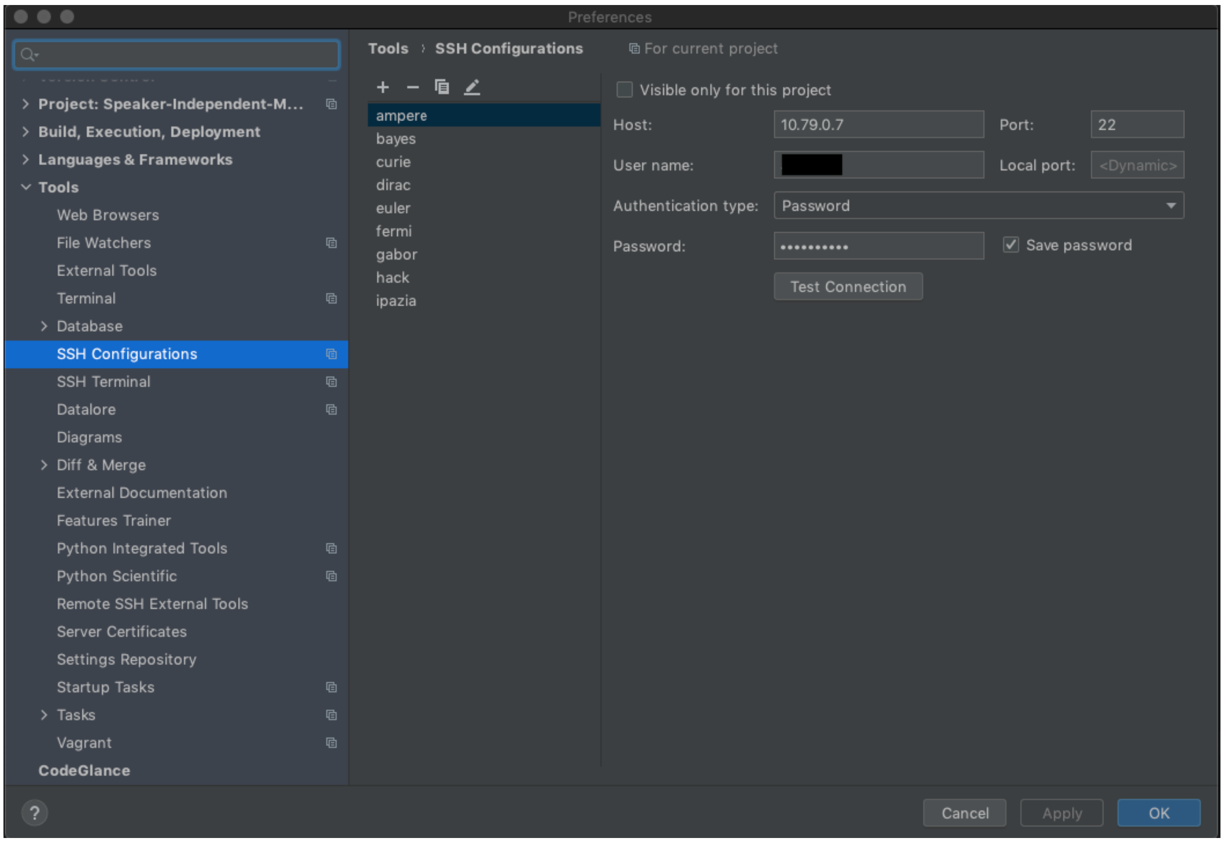
Its mechanism can be described as follows:

- PyCharm connects to the machines via SSH using your user account
- Projects are edited on your local machine, and automatically uploaded and synced with the remote host; this operation is called **deployment**
- A **remote Python interpreter** must be created through a SSH host, pointing to your *Conda* env.

Preliminary steps:

0. Install PyCharm on your PC and open it
1. Open the "*Preferences…*" panel, a window will show up
2. Go to "*SSH Configuration*" panel (eitbayesher search for it, or find it under *"Tools"* group)
3. For each remote machine, create a connection configuration using the + button and the *SFTP* option:
    a. **Host:** <IP_ADDRESS>
    b. **Port:** 22
    c. **User name:** <YOUR_USER_NAME>, i.e. *mrossi*
    d. **Local port:** <Dynamic>
    e. **Authentication type:** either the SSH key id_rsa you created following this guide (suggested), or the user password
    f. Test the connection to check that everything is **OK!**
    g. For your convenience, rename the configuration using the name of the machine (e.g., *ampere*) by clicking on the pencil button
    h. Apply the changes.

    At the end, you will have a similar window:



Visible only for this project: ☐

Host: 10.79.0.7 Port: 22

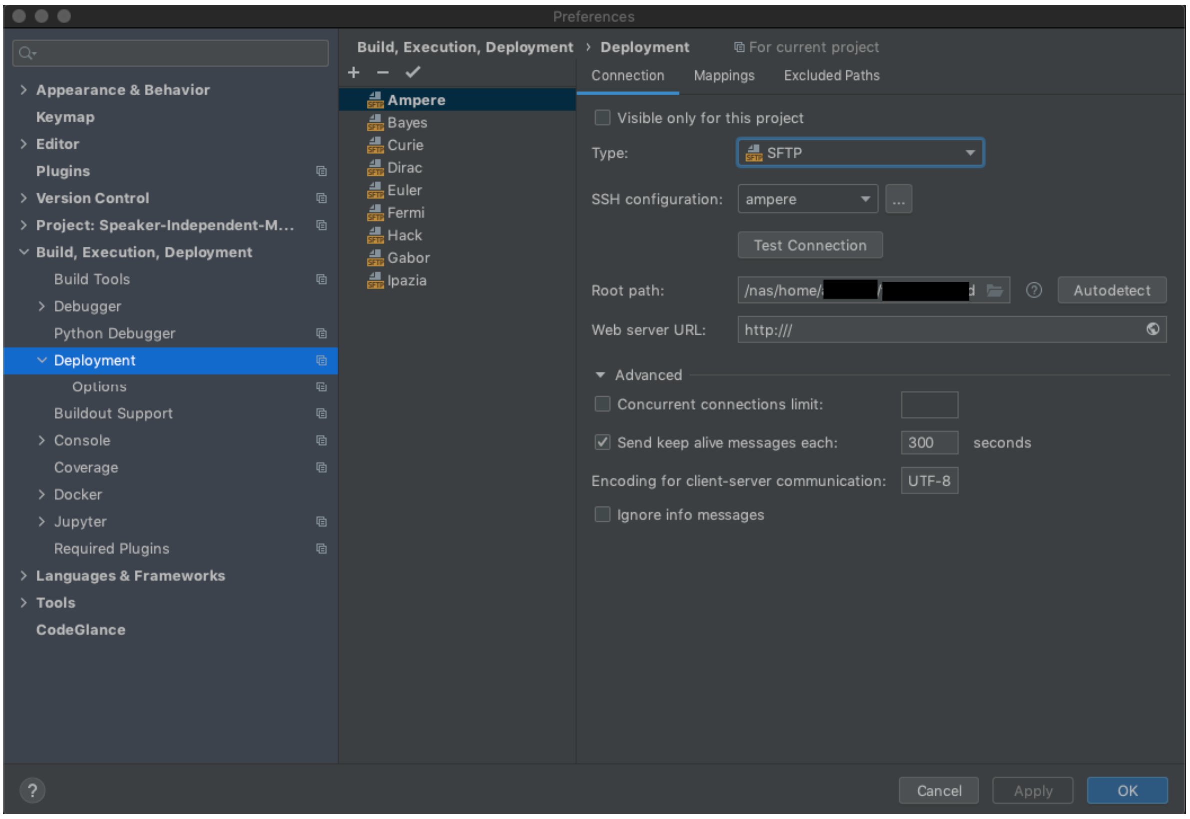
User name: [redacted] Local port: <Dynamic>

Authentication type: Password

Password: •••••••••• ✓ Save password

Deployments configuration:

1. Open the "*Preferences…*" panel, go to *"Build, Execution, Deployment → Deployment"*
2. It is strongly suggested to create a default deployment configuration to *Ampere*, using the + button:
   a. **Type:** SFTP
   b. **Name:** for your convenience, use the machine name with the capital letter (e.g., Ampere)
   c. Untick the "visible only for this project" box
   d. **SSH configuration:** select the "ampere" ssh configuration from the drop-down menu
   e. **Root path:** */nas/home/mrossi*
   f. **Web server URL:** leave blank



Visible only for this project: ☐

Type: SFTP

SSH configuration: ampere

Root path: /nas/home/[redacted]

Web server URL: http:///

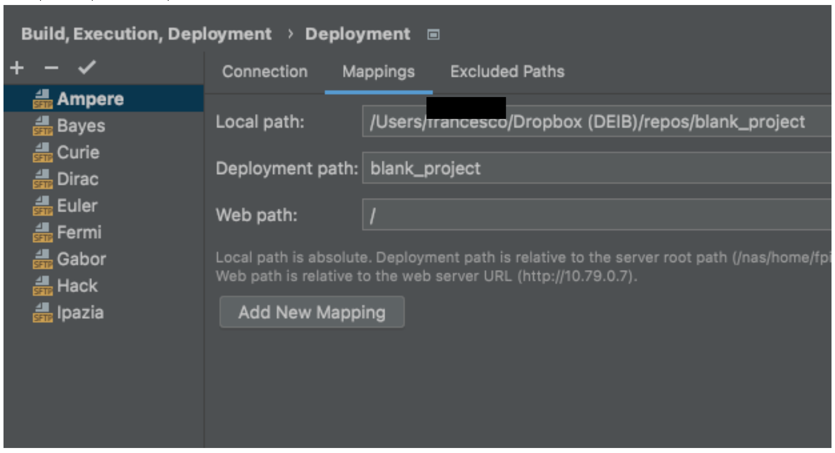
Concurrent connections limit: ☐

Send keep alive messages each: ✓ 300 seconds

Encoding for client-server communication: UTF-8

Ignore info messages: ☐

3. Go to the *"Mappings"* tab above, then:
   a. **Local Path:** the path pointing to your project folder
   b. **Deployment path:** name of the remote project folder in the remote folder, without the*/nas/home/mrossi* folder path preamble.



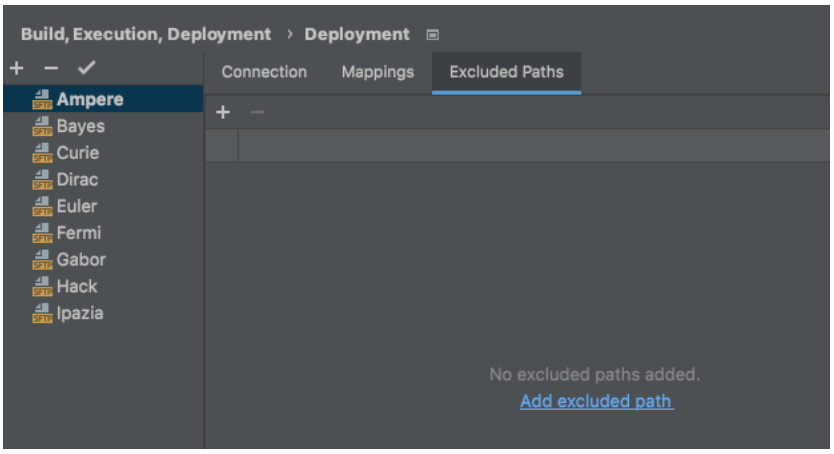
Local path: /Users/[redacted]/Dropbox (DEIB)/repos/blank_project

Deployment path: blank_project

Web path: /

Local path is absolute. Deployment path is relative to the server root path (/nas/home/fpi
Web path is relative to the web server URL (http://10.79.0.7).

4. If needed, you can exclude some local and remote folders (i.e. local papers folder and remote results folders) in the *"Excluded Paths"* tab



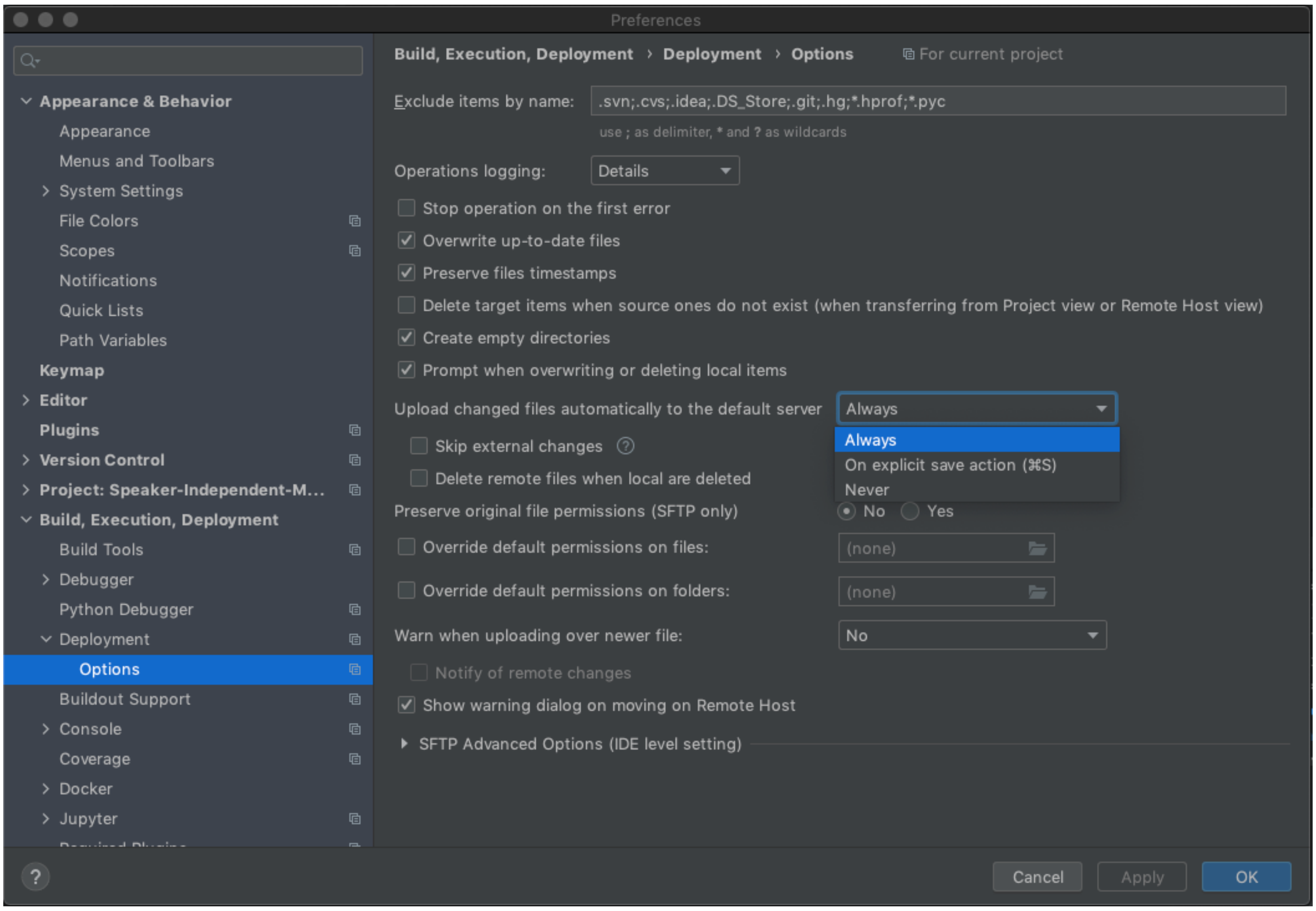
No excluded paths added.

5. When done, click *"Apply"* in the bottom right of the window.

6. It is strongly suggested to create a default deployment configuration to the *Ampere* workstation by clicking the ✓ button when selected. The workstation's name will become bold. This default deployment will be used for automatic uploading.

Additional configuration

Open the "*Preferences…*" panel, go to *"Build, Execution, Deployment → Deployment → Options"*.



Preferences

- Appearance & Behavior
  - Appearance
  - Menus and Toolbars
  - System Settings
  - File Colors
  - Scopes
  - Notifications
  - Quick Lists
  - Path Variables
- Keymap
- Editor
- Plugins
- Version Control
- Project: Speaker-Independent-M…
- Build, Execution, Deployment
  - Build Tools
  - Debugger
  - Python Debugger
  - Deployment
    - Options
  - Buildout Support
  - Console
  - Coverage
  - Docker
  - Jupyter

Build, Execution, Deployment › Deployment › Options   For current project

Exclude items by name: .svn;.cvs;.idea;.DS_Store;.git;.hg;*.hprof;*.pyc
use ; as delimiter, * and ? as wildcards

Operations logging: Details

[ ] Stop operation on the first error
[✓] Overwrite up-to-date files
[✓] Preserve files timestamps
[ ] Delete target items when source ones do not exist (when transferring from Project view or Remote Host view)
[✓] Create empty directories
[✓] Prompt when overwriting or deleting local items

Upload changed files automatically to the default server: Always
- Always
- On explicit save action (⌘S)
- Never

[ ] Skip external changes
[ ] Delete remote files when local are deleted

Preserve original file permissions (SFTP only): [✓ No] Yes

[ ] Override default permissions on files: (none)
[ ] Override default permissions on folders: (none)

Warn when uploading over newer file: No

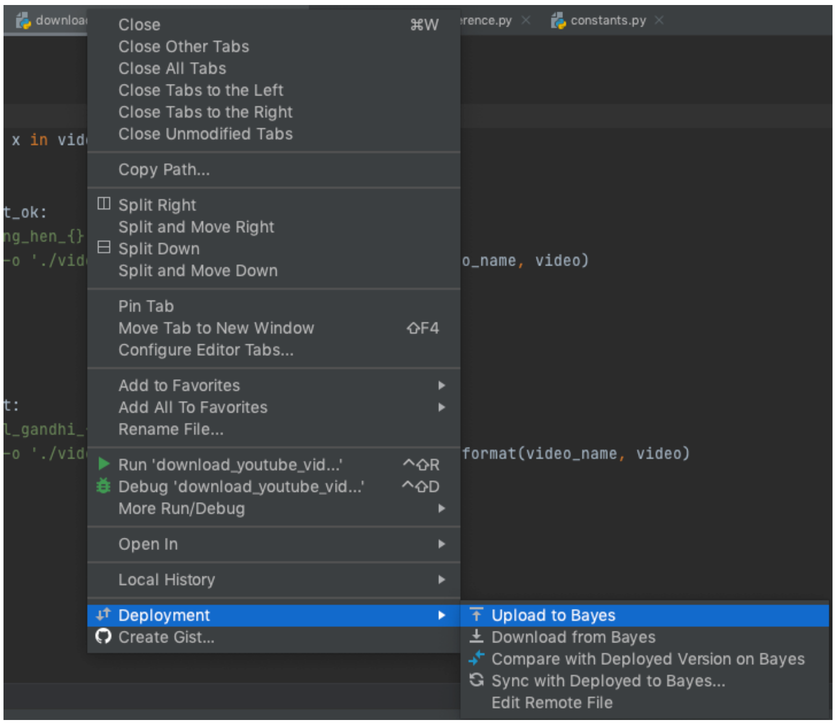
[ ] Notify of remote changes
[✓] Show warning dialog on moving on Remote Host

SFTP Advanced Options (IDE level setting)

Cancel   Apply   OK

Here you can set some additional option regarding the automatic upload and some general upload options.
A you can notice, you can select different modalities for the automatic upload, also some options related to the remote backup behavior, like the overwrite operation on up-to-date files.

You can force the upload on the remote server by right click on the local Python file that you are editing. Under the *"Deployment"* submenu you can make some operation related to the remote workstation, as you can see below.

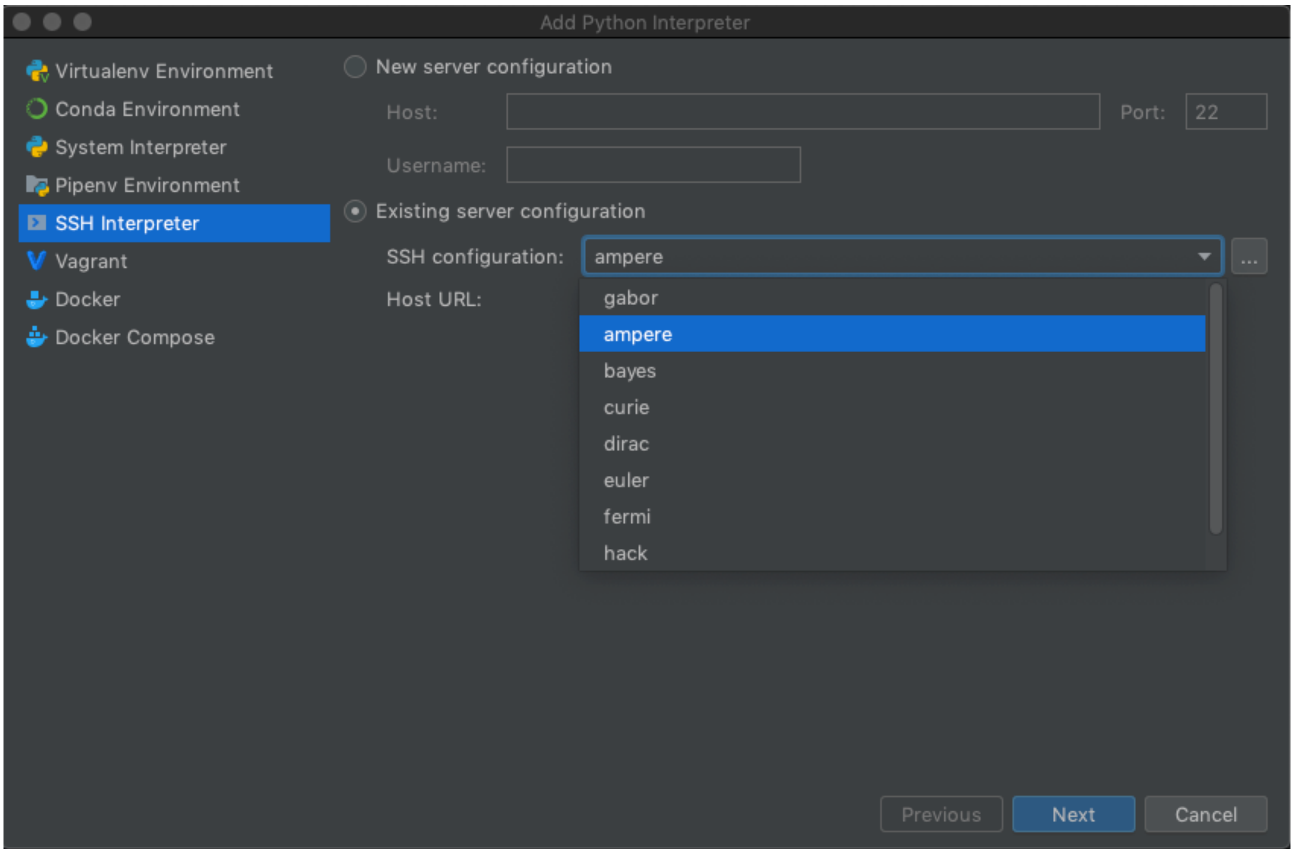


In this case, he selected remote workstation is the one set in point 6. of the "*Deployments configuration"* section above.

Remote Interpreter:

Once you have done with the deployment configuration, you must configure the workstation to work with your project, connecting the proper *Conda* environment.

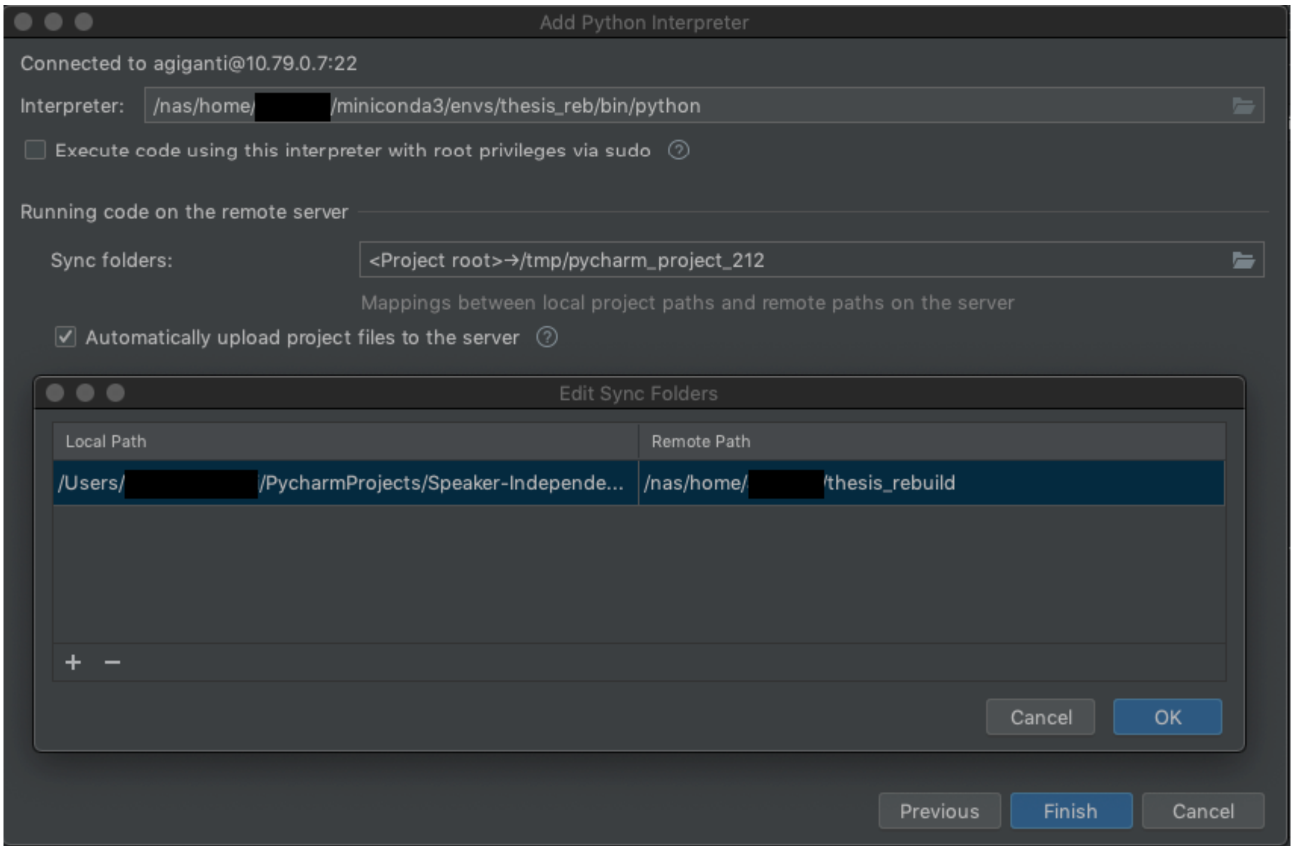
1. Go to "*Preferences* → *Project:<YOUR_PROJECT>* → *Python Interpreter"*
2. Click the ⚙ icon and then *"Add…"*
3. Got to the *SSH Interpreter* submenu and tick the "*Existing Server Configuratio*n" option
4. In the dropdown menu, select the SSH Configuration previously configured and click *"Next"*



5. Now, by clicking the 📂 icon on the "*Interpreter:"* field, you have to select the correct Python executable file, related to a **specific Conda** environment.

The final path for the python interpreter will be something like this:
/nas/home/mrossi/miniconda3/envs/<ENV_NAME>/bin/python

6. By clicking the icon on the "*Sync folders:*" fields instead, you can map your remote project folder with your local one. The final path for the mapping will be something like this:
/nas/home/mrossi/<PROJECT_NAME>, but you can choose another folder in your personal folder, i.e. the standard *mrossi*.



Connected to agiganti@10.79.0.7:22

Interpreter: /nas/home/[redacted]/miniconda3/envs/thesis_reb/bin/python

☐ Execute code using this interpreter with root privileges via sudo

Running code on the remote server

Sync folders: <Project root>→/tmp/pycharm_project_212

Mappings between local project paths and remote paths on the server

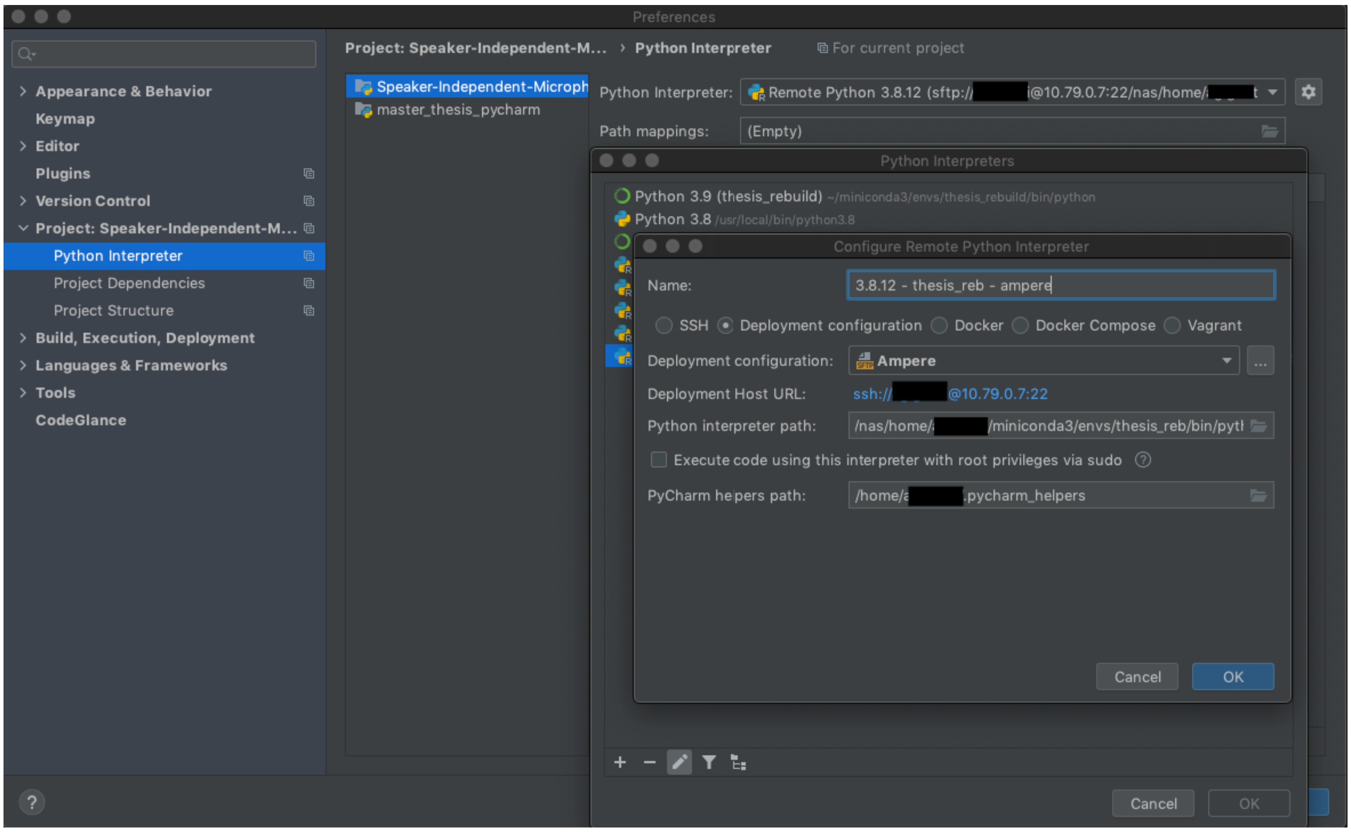
☑ Automatically upload project files to the server

Edit Sync Folders

Local Path	Remote Path
/Users/[redacted]/PycharmProjects/Speaker-Independe...	/nas/home/[redacted]/thesis_rebuild

If you want PyCharm uploads automatically your files on the remote folder after each local changes, you can tick the "*Automatically upload project files to the server*". This operation overwrites the previous remote version of the files in that folder.

7. You can rename your environment. In "*Preferences → Project:<YOUR_PROJECT> → Python Interpreter*", click the icon and then *"Show all..."*. By clicking the icon in the bottom part, you see a window like this. In this, you can change the name that will be displayed in the bottom part of the PyCharm IDE.
This operation is useful when you have different interpreters related to different project/environments.



Python Interpreter: Remote Python 3.8.12 (sftp://[redacted]@10.79.0.7:22/nas/home/[redacted]t

Path mappings: (Empty)

Python 3.9 (thesis_rebuild) ~/miniconda3/envs/thesis_rebuild/bin/python

Python 3.8 /usr/local/bin/python3.8

Name: 3.8.12 - thesis_reb - ampere

SSH [✓ Deployment configuration] Docker Docker Compose Vagrant

Deployment configuration: Ampere

Deployment Host URL: ssh://[redacted]@10.79.0.7:22

Python interpreter path: /nas/home/[redacted]/miniconda3/envs/thesis_reb/bin/pytl

☐ Execute code using this interpreter with root privileges via sudo

PyCharm helpers path: /home/a[redacted].pycharm_helpers

NB. Each configuration done in this section refers to a single Python **interpreter**. A single interpreter is related to a specific *Conda* environment. If you want to execute the code on all the remote machines, you should make **one configuration for each python executable** related to a specific environment and **for each workstation** that you plan to use.