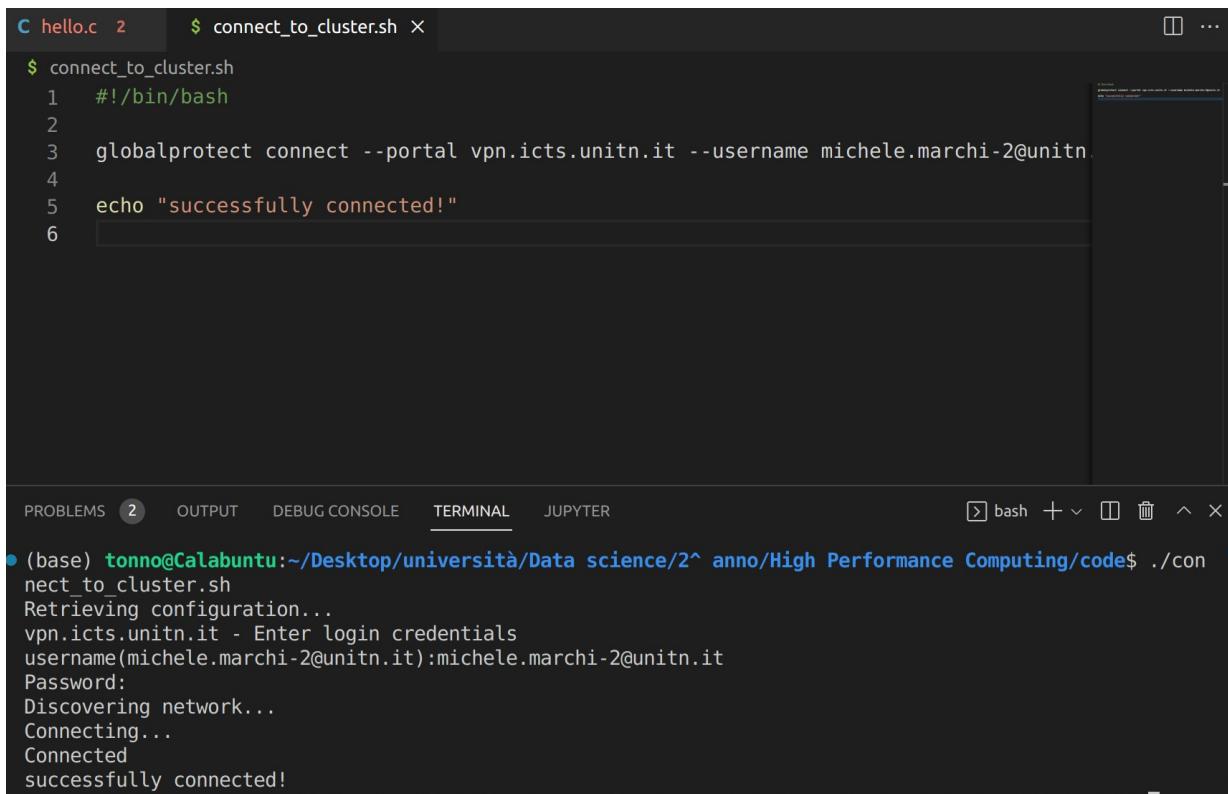




SSH in VS Code

Connect to the VPN

With Linux you can also do from the command line



The screenshot shows a terminal window within a code editor interface. The terminal tab is active, displaying the command `connect_to_cluster.sh`. The script content is:

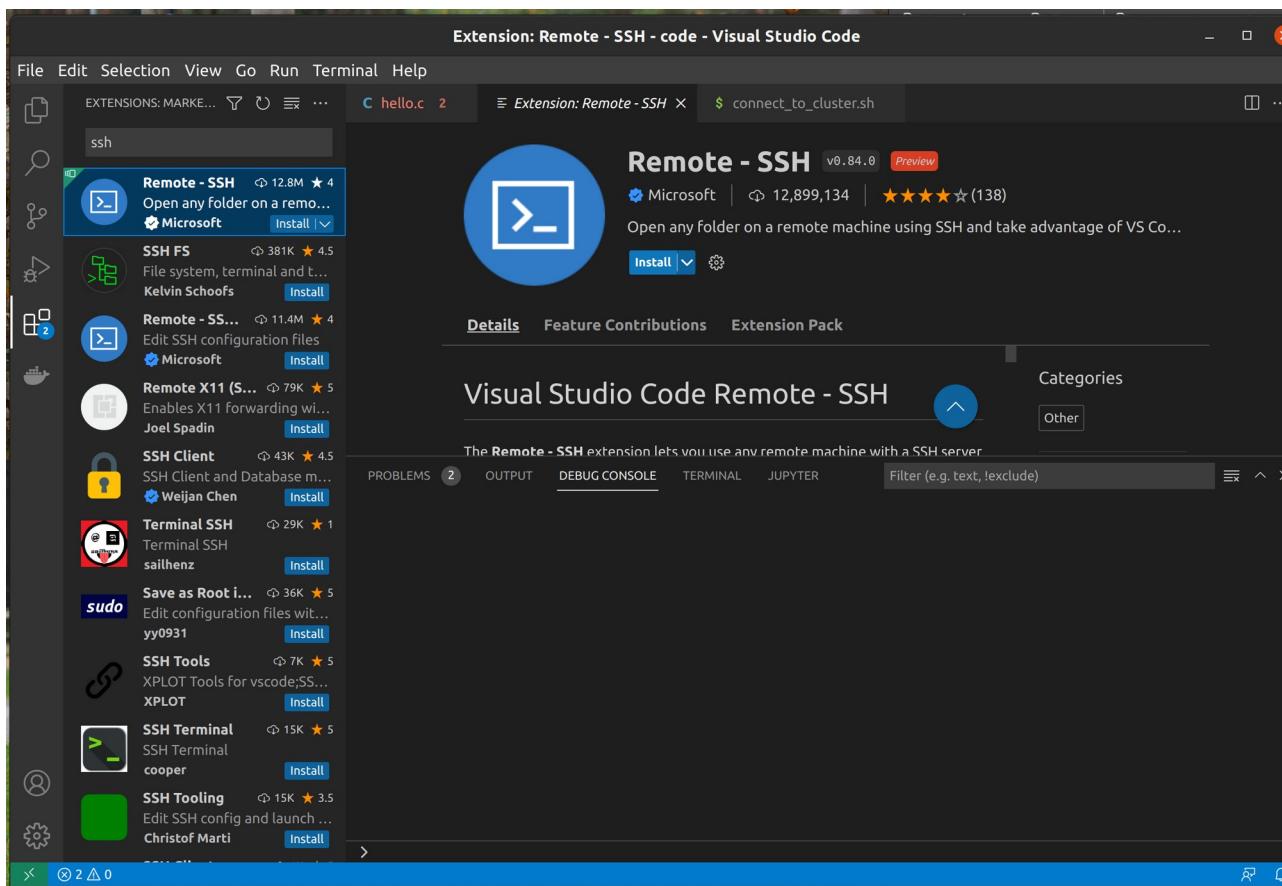
```
$ connect_to_cluster.sh
1  #!/bin/bash
2
3  globalprotect connect --portal vpn.icts.unitn.it --username michele.marchi-2@unitn.
4
5 echo "successfully connected!"
```

Below the terminal, the output of the script execution is shown:

```
(base) tonno@Calabuntu:~/Desktop/università/Data science/2^ anno/High Performance Computing/code$ ./connect_to_cluster.sh
Retrieving configuration...
vpn.icts.unitn.it - Enter login credentials
username(michele.marchi-2@unitn.it):michele.marchi-2@unitn.it
Password:
Discovering network...
Connecting...
Connected
successfully connected!
```

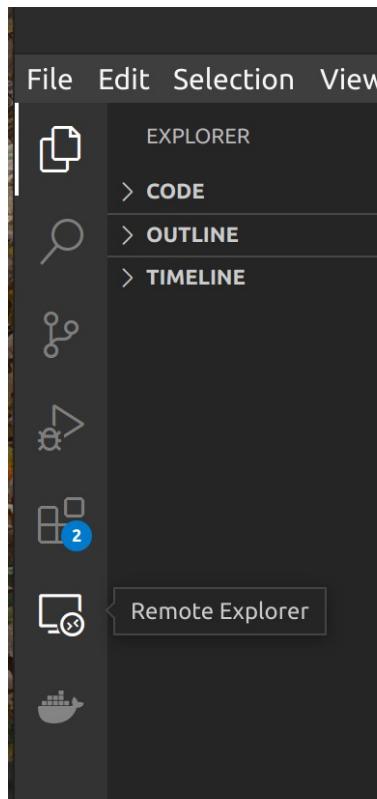
Install SSH extension

Go in the extention menu and search “SSH”, the first result is just fine



Install SSH extension

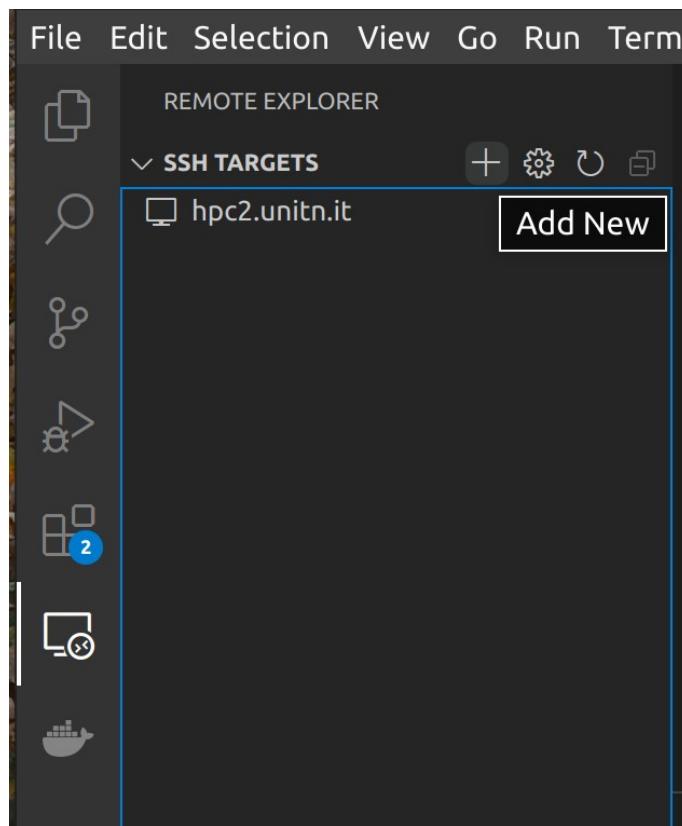
After the installation a new icon will appear on the left side.



Connect with SSH

After the installation a new icon will appear on the left side.

- Open it and press on “Add New” icon;
- Write the details of the SSH connection on the search bar, saving the configuration wherever you wish.



Connect with SSH

You should now receive a message on the lower right side.

Now open the connection of the “SSH target” in a New Window

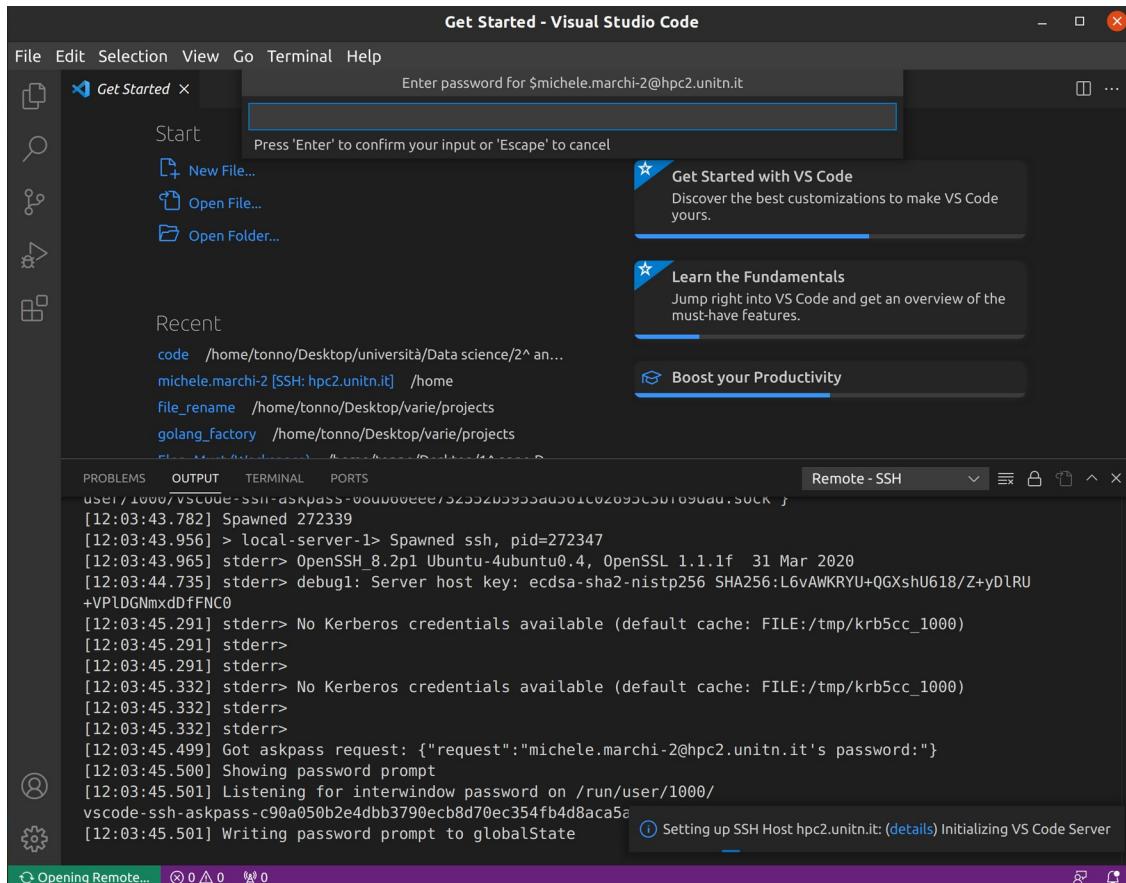
The screenshot shows a Visual Studio Code interface with the following details:

- Title Bar:** hello.c - code - Visual Studio Code
- File Menu:** File Edit Selection View Go Run Terminal Help
- Remote Explorer:** REMOTE EXPLORER
- SSH Targets:** hpc2.unitn.it
- Code Editor:** C hello.c 2 X \$ connect_to_cluster.sh
Content:

```
hello_world > C hello.c > main(int, char **)
1 #include <mpi.h>
2 #include <stdio.h>
3
4     // Initialize the MPI environment. The two arguments to MPI Init are not
5     // currently used by MPI implementations, but are there in case future
6     // implementations might need the arguments.
7     MPI_Init(NULL, NULL);
8     // Get the number of processes
9     int world_size;
10    MPI_Comm_size(MPI_COMM_WORLD, &world_size);
11    // Get the rank of the process
12    int world_rank;
13    MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);
14    // Print off a hello world message
15    printf("Hello world from process rank %d out of %d processors\n", world_rank);
```
- Terminal:** bash + ↻
- Output:** bash + ↻
- Message Bar:** Host added! (Source: Remote - SSH (Extension))
- Status Bar:** Ln 18, Col 2 Spaces: 4 UTF-8 LF C Linux
- Page Number:** 6 / 9

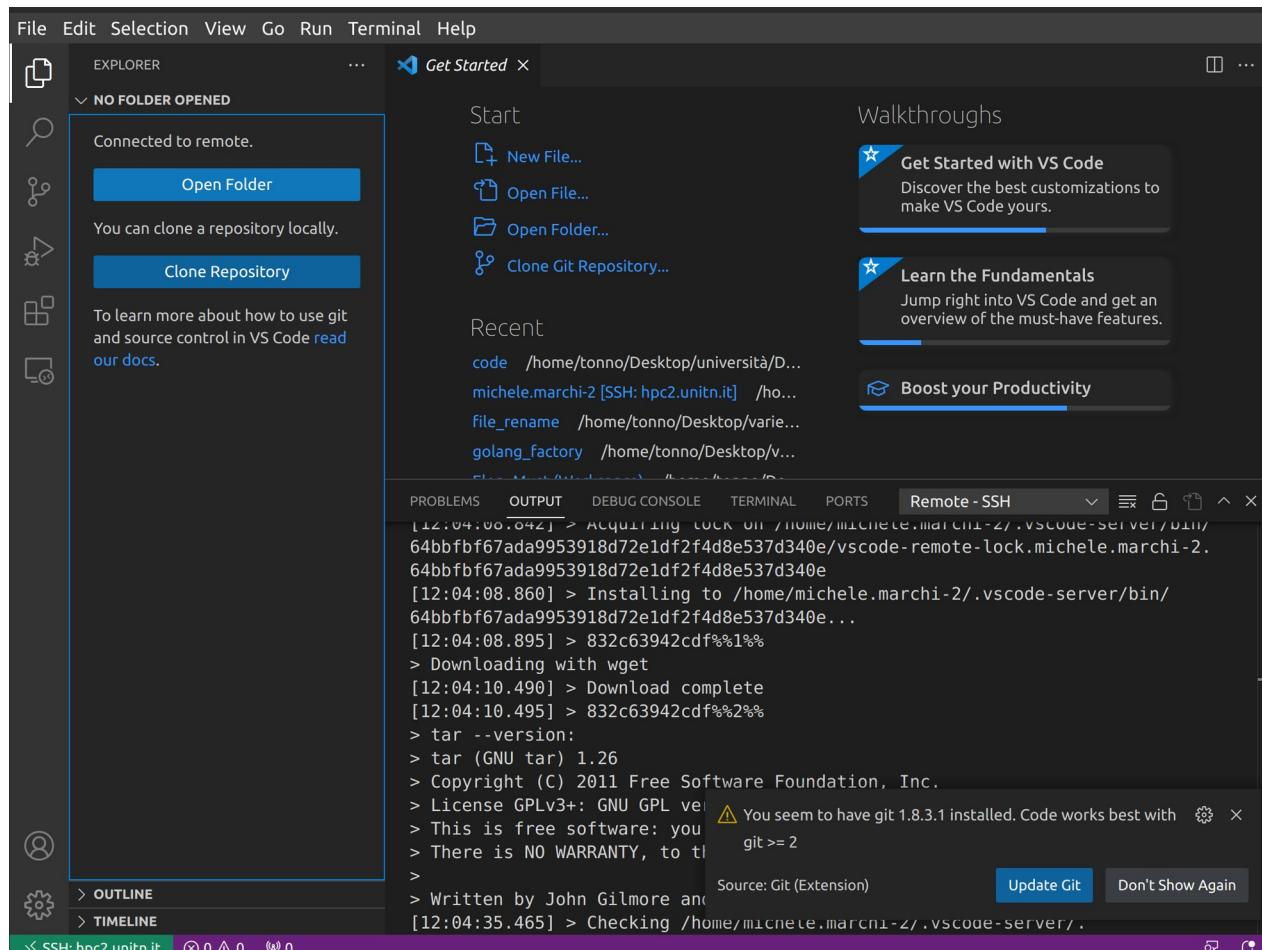
Inside the cluster

In the new window you are requested to do the login again



Inside the cluster

You are now in and able to open your cluster's folder, after inserting your credentials again.



Congratulations

You can now play with the support of the GUI!

The screenshot shows a Visual Studio Code interface running on a Linux system. The title bar reads "pingpong.sh.o8894728 - michele.marchi-2 [SSH: hpc2.unitn.it] - Visual Studio Code". The left sidebar shows a file tree with a folder named "MICHELE.MARCHI-2 [SSH: HPC2.UNITN.IT]" containing files like ".cache", ".config", ".vscode-server", "01_hello", "02_circle", "03_pingpong", "pingpong", "pingpong.c", "pingpong.sh", "pingpong.sh.e8894728", "pingpong.sh.o8894728", ".bash_history", ".bash_logout", ".bash_profile", ".bashrc", ".emacs", and ".viminfo". The main editor area displays the contents of "pingpong.sh.o8894728", which contains a script that prints "I am 1, and I pong" and "I am 0, and I ping" followed by a sequence of "whoosh" and "whoop" words. Below the editor is a tab bar with "PROBLEMS", "OUTPUT", "DEBUG CONSOLE", "TERMINAL", and "PORTS", with "TERMINAL" being the active tab. The terminal pane shows the command "[michele.marchi-2@hpc-head-n2 ~]\$". The bottom status bar indicates the file is saved with "pingpong.sh.o8894728".