Setup your VM on surf research cloud

R. Grouls
Version 1.0

September 8, 2025

Contents

Chapter 1

Create a VM

1.1 Introduction

At the MADS, we will be using the SURF Research Cloud to run virtual machines. This has a lot of advantages:

1. Uniformity of hardware and software, so we as teachers can better assist you.
2. Lesson material is tested on the operating system of the VMs, so it is guaranteed to work.
3. Instructions are optimized for the VMs, so you can follow them step by step.
4. We will have GPUs available on the VMs, so you can run deep learning models with considerably less time.
5. It is pretty common to run code on a VM in production; getting familiar with setting up VMs is a useful skill to have.

Find the invite

This manual will guide you through the process of setting up your VM on the SURF Research Cloud. The first requirement is that you search in your @student.han.nl inbox for an inviation to join the collaboration of your year. We use UOS1, UOS2 and UOS3 for the different semesters. If you have not received an invite, please contact us.

1.2 Finding your VM

Click on the invite link in your email. Log in with your @han email account. You should see something like you see in fig 1.1, and you should see that you collaborate in your UOS group.

If you access the collaboration, you go to the dashboard (see fig 1.2). We already created a VM for you. Typically, it will be:

1. An ubuntu machine with 8GB RAM (good for UOS1 and UOS2)
2. An ubuntu machine with 16GB RAM (good for UOS3)
3. An ubuntu machine with a GPU (good for the hackathon and UOS3)

Find your machine

We created the VMs with UOSx-FLastname, where x is the number of your UOS, and F is the first letter of your Firstname. You can use the search function to find the machine with your name.

If you ever are in a situation where you need to decide for yourself what type of machine you need, some rough guidelines:

1. For the processing of small datasets (most datasets we use are considered small) 8GB is enough. Note that the pandas library has a really unhealthy relationship to RAM usage; often it is usefull to switch to the more modern polars library for preprocessing if you run into RAM problems because of pandas.

2. For most data visualisation, 8GB RAM is enough

3. For most simple programs in deployment, 8GB (or even less) is often more than enough to run the program.

4. When doing machine learning, some models require more RAM. Typically, image models (CNN architectures) require more RAM, typically about 16GB. The same for Transformer architectures for the bigger models, and most LLMs will need 16GB or even more, depending on the size of your LLM.

5. Inference (making a prediction) for classic Neural Networks is often fast enough on a CPU. However, if the models get really big (like they tend to be with the modern Transformer architectures used for LLMs) you probably need a GPU even for inference.

1.3 Setup your VM

As a rule of thumb: it is smart to make it easy to setup your own VM, such that setting up a new VM is just a few minutes of work. This means that you should:

1. Use git to store your code, and push your code to github/gitlab.

2. Use scripts to install the things you need on a VM

Take a look at my repo `https://github.com/raoulg/serverinstall`; I use these scripts to install things I like with a oneliner. For example, if i need docker, i figured this out with the docker manual once, and created this `https://github.com/raoulg/serverinstall/blob/master/install-docker.sh` script. Now all i need to do is run the curl command you can find in the readme of my serverinstall repo on the new machine, and my script is downloaded with curl and executed with bash.

Why go to this trouble? Because with a VM, you can start with an 8GB machine, once you run into issues you can scale up within minutes to a 16GB or even 32GB machine, and if you want to train your model you spin up a GPU.

It is very common to write code locally or on a small machine, and only spin up a GPU VM at the moment you need it and simple ‘git clone‘ your work to the GPU machine.

1.4 SSH key

You will need a SSH key. If you don't already have one and need a refresher on how to create one, see the following link: `https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent`.

If you don't really have an idea what an SSH key is, please watch this 10 minute video `https://youtu.be/dPAw4opzN9g?si=3k8Rvvd5uYgeQEfd`. Also, if you decide to skip it, but you get all sorts of issues with the ssh-keys, it is probably because you dont understand what you are doing. So the 10 minutes is a good investment of your time, you will earn it back!

You should add your public key to the workspace setup.

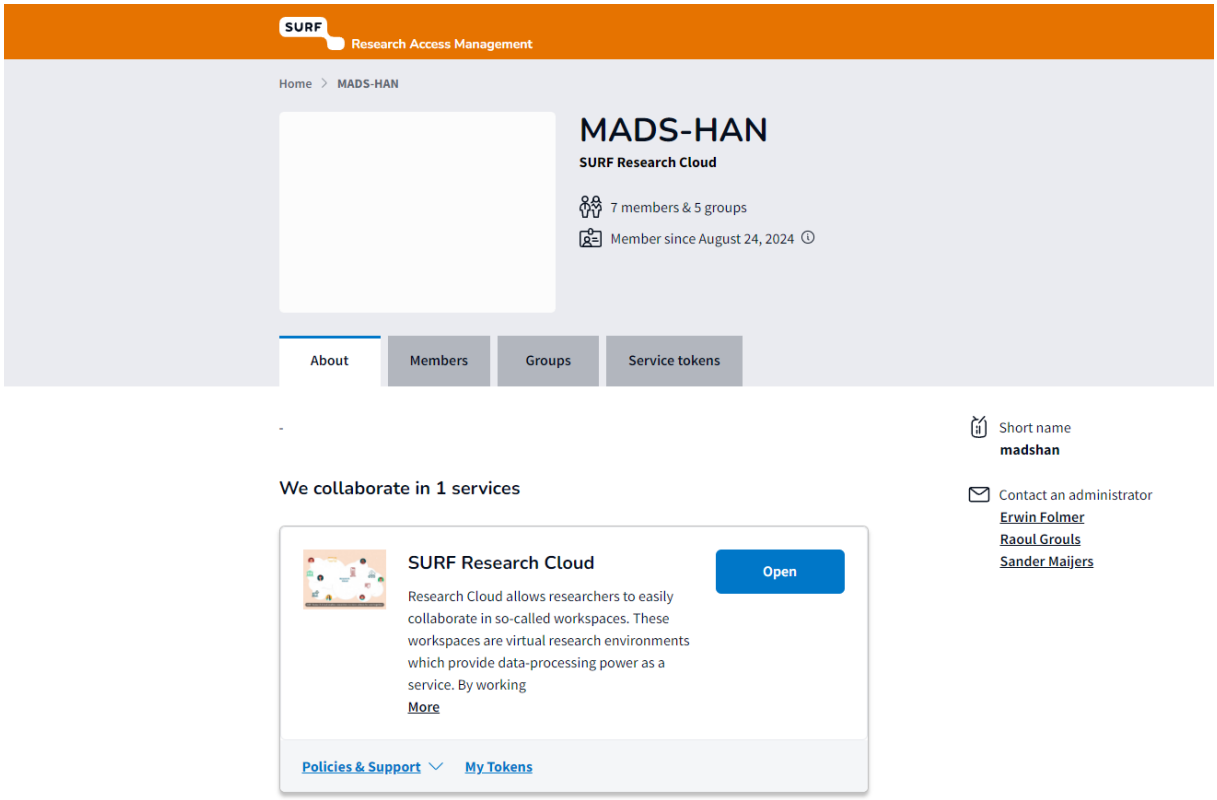

Figure 1.1: Open the SURF Research Cloud

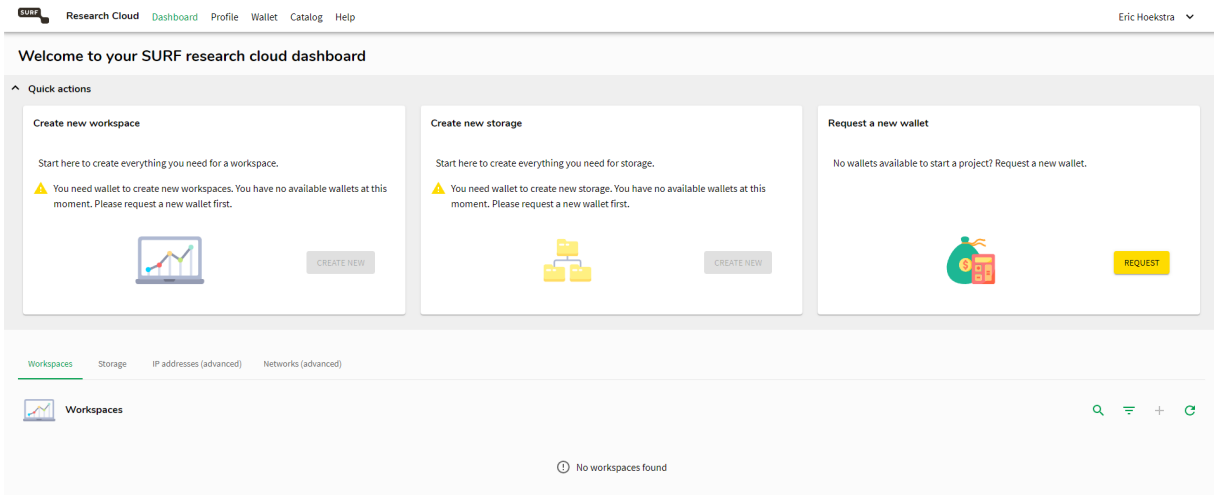

Figure 1.2: SURF dashboard

The SSH keys

1. Please, pay attention **where** you store your keys on your system. If necessary, write down the location. On linux and mac it will typically be in ~/.ssh, on windows it can be anywhere depending on your setup. Your public key is a string of characters inside a text file; if you open the .pub file with a text editor, you should see something like `ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDQ...`. This is what you need to copy-paste into the SURF Research Cloud.

2. Never share your private key, only the public key (ending in .pub).

If you go to your profile in the dashboard, you should be able to find a tab "public ssh keys". This is where you store your public ssh key; it will be copied to your machine automatically.

Chapter 2

Connect to your VM

2.1 Pausing and Resuming your VM

Running a small VM (2 cores, 16GB) will cost you about 50 cents per hour. For a GPU, you typically pay closer to 5,- per hour. This means that you can just let your VM run forever; you are paying about 120,- per day if you dont shut down your GPU, and when 30 student do this, a class would cost 108.000,- euros per month, which is too much.

Please note that even if you don't actually use the VM, we still pay for it: it is still "on", regardless if you are logged in. So, when you are not using the VM, **please pause** it. You can do this in two ways:

1. Log in to the SURF Research Cloud, navigate to your Dashboard, and click "Resume" or "Pause" on your workspace.

2. Use surfcontroller, a package we developped in Python. You can install it globally by running `uv tool install surf-controller`. Because the package is published on pypi (see `https://pypi.org/project/surf-controller/`). You can then run `surfcontroller` from your terminal and select your VMs to pause/resume, see Figure 2.1.

Autoshutdown @ 21:00

1. We will **automatically pause all VMs every day at 21:00**. Experience learns that there will always people who forget to shutdown their VM, so we rather have the small inconvenience of the autoshutdown at 21:00 than to have to constantly check if someone forgot to shut down his machine.

2. If you want to do computations that need more time (eg for hypertuning your model) you can request a lift from the autoshutdown with your teacher. Log in to surf dashboard and copy the ID of your machine to your teacher with the request.

2.2 Connecting to your VM

After the VM is created, you can use the dropdown menu to find the URL with the IP address of your VM. You can use this IP address to connect to your VM, see Figure 2.4. Make sure your VM is running before you try to connect. You can open VScode and when you have installed the "Remote Development" extension from Microsoft, you will have a small blue icon in the bottom left corner, see Figure 2.2. Click on it and choose "Remote-SSH: Connect to Host...", see Figure 2.3. You can now enter `user@IP-address` of your VM and connect to it. You will find this under 'URL' in the dropdown menu. For example, I would enter `rgrouls@145.38.192.86` to connect.

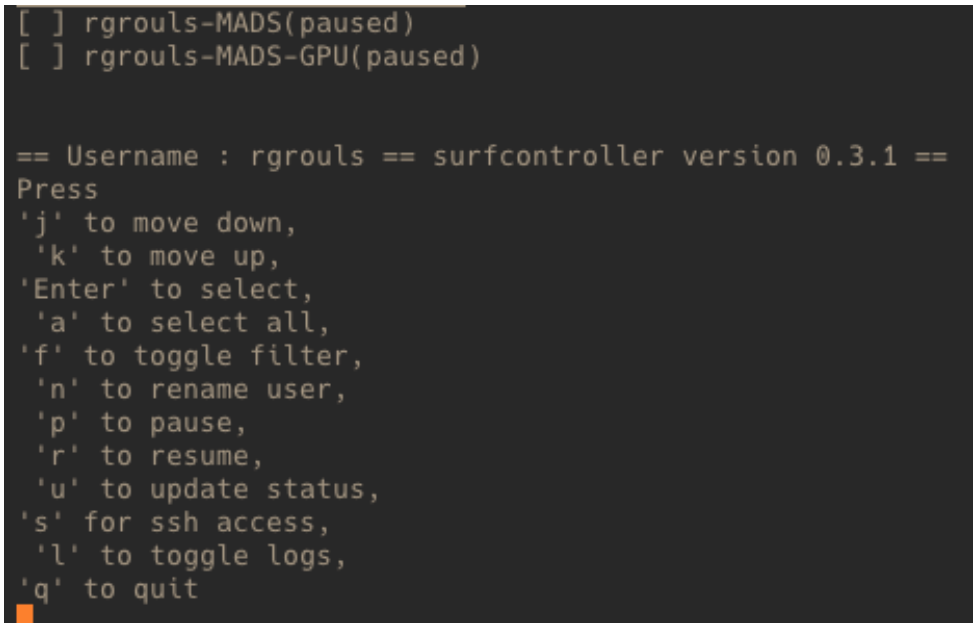

Figure 2.1: Surfcontroller screenshot

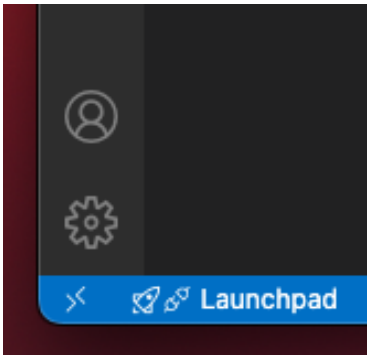

Figure 2.2: Click the small blue icon in the bottom left corner

The first time you will get a question saying this is an unknown host and if you want to add its fingerprint. You can accept this and continue. After this, you should have entered your VM. You can now start working on your VM.

2.2.1 SSH without password

If you still need to enter a password when connecting with VScode, you might need check your SSH key; some people create their SSH key with a password. It is fine to recreate one without a passworkd and add it to your SURF profile.

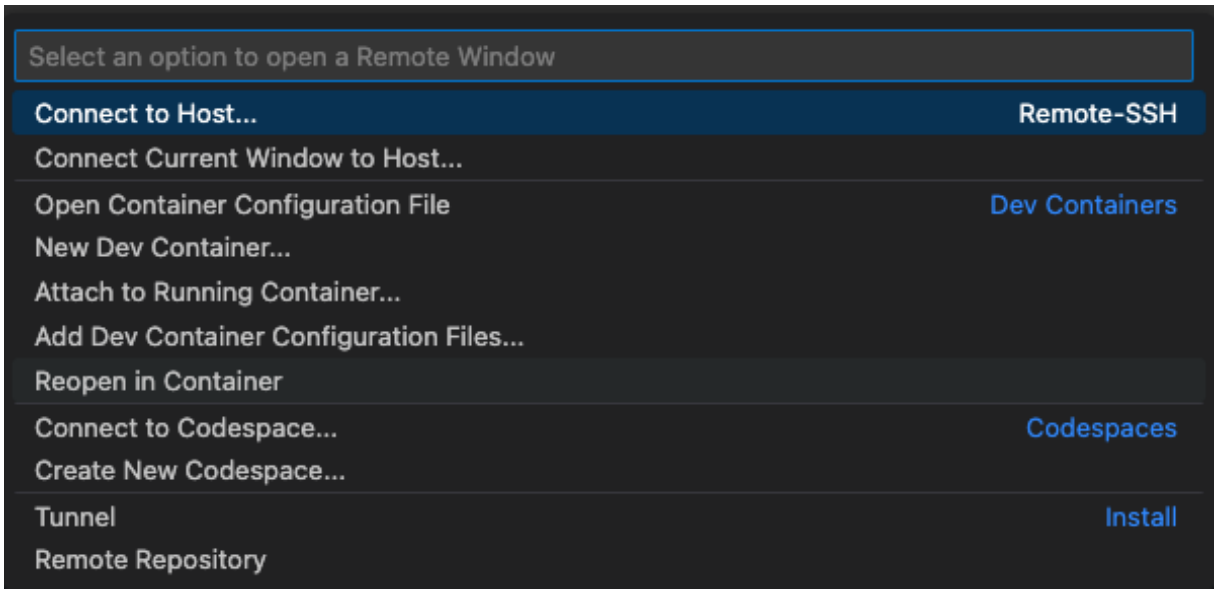

Figure 2.3: connect to host

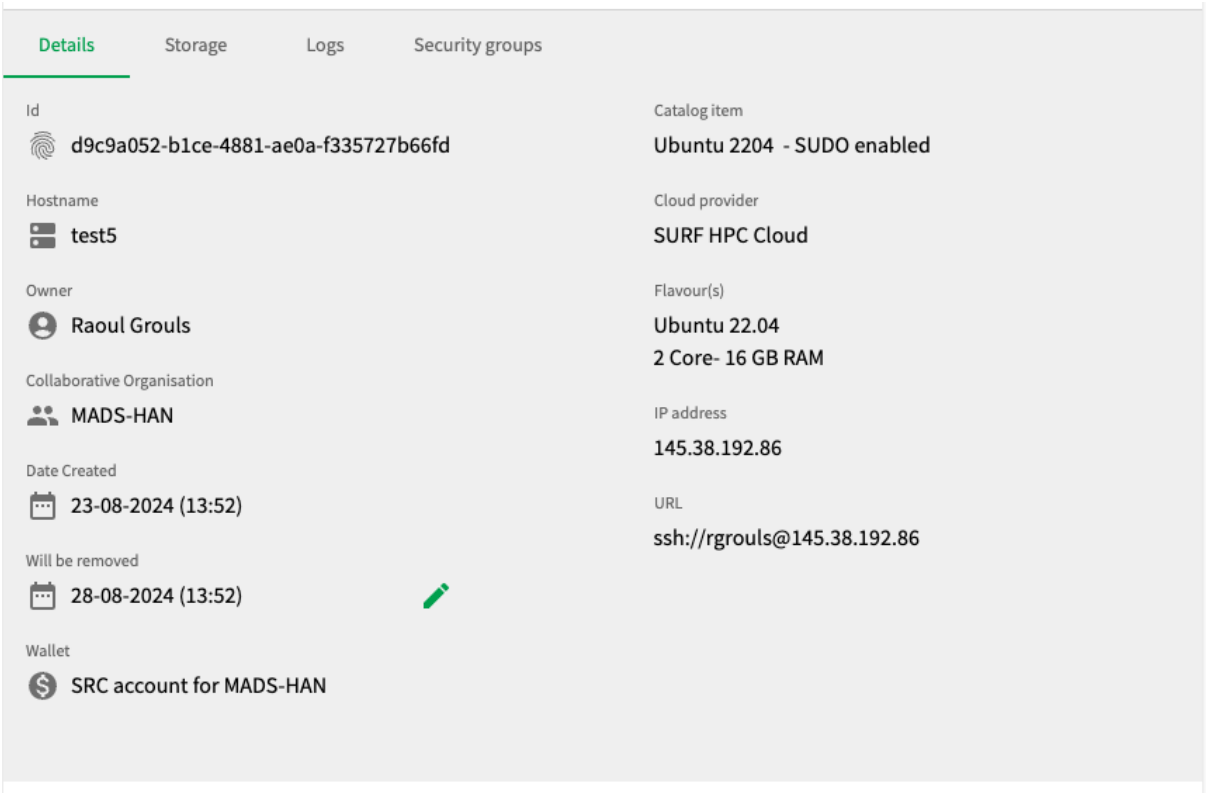

Figure 2.4: Find you IP address and url

Chapter 3

Install the course

3.1 Clone the github repo

In the `README` of the course `https://github.com/raoulg/MADS-MachineLearning-course`, you can find the instruction to 1. install uv. This will manage python and the dependencies. 2. clone the repo 3. install the requirements with `uv sync`

3.2 Using the VM and Git

You will use the VM as a backend for your computing power.

Make your VM disposable

Don't use your VM as a storage of your work. Sometimes this happens without you realising you do:

1. modifying notebooks with their personal things and NOT PUSHING these modification to github
2. running notebooks and using the output of the notebook as some sort of outoput storage

In the first case: it is fine to fork the repo, and in your own repo branch off changes you want to make to notebooks. As long as you make sure you push the changes to git, you are good.
In the second case: this is a bad practice. Never use jupyter notebooks as storage; but save images to disks, export trained models to weights, etc and store them in the `/data` folder in your repo.

Chapter 4

TLDR

TLDR

please read all the boxes similar to these and you're good.