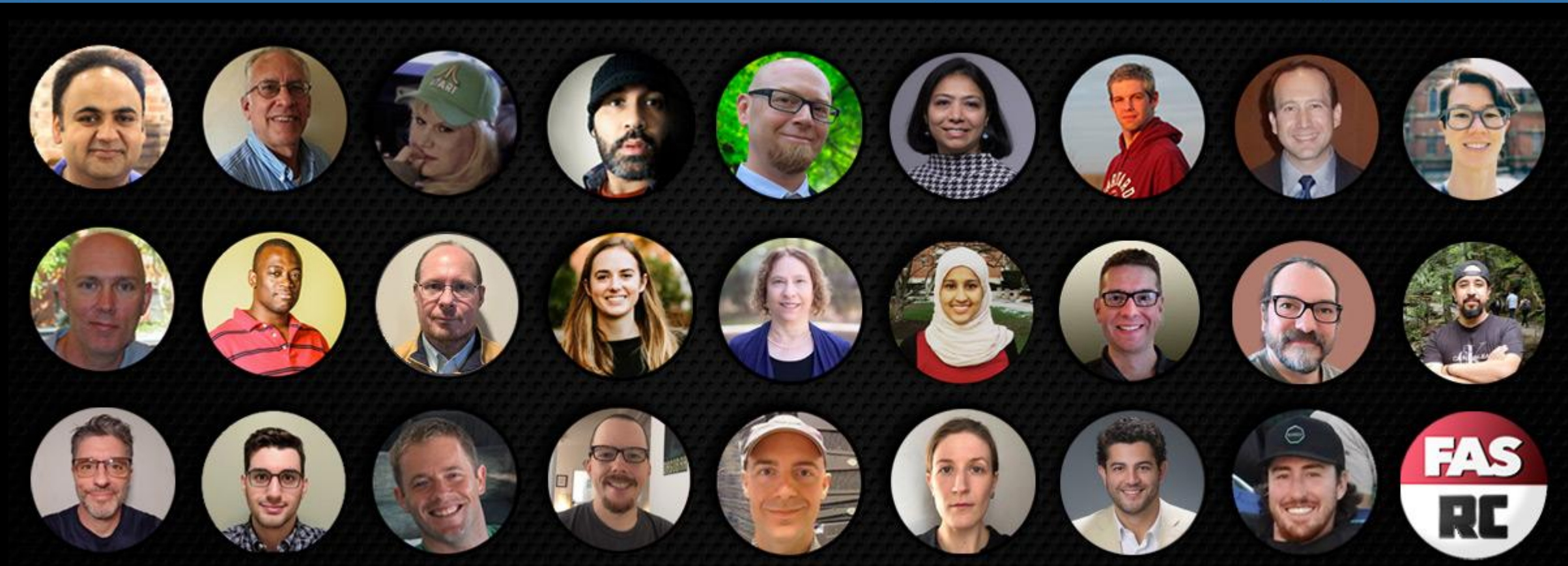


VSCode on the FASRC clusters

Learning objectives

- What is VSCode
- Pros & Cons
- Ways to launch VSCode on Cannon
  - Remote Tunnel with Sbatch
  - Remote SSH via ProxyCommand
- Best Practices
- Issues to look out for
- VSCode Basics & Other Considerations
- Cursor Bits

VSCode

- Microsoft Visual Studio Code, VS Code, most popular code editor
- Source-code editor developed by Microsoft for Windows, Linux, macOS and web browsers
- Features include:
  - support for debugging, syntax highlighting,
  - intelligent code completion, snippets management, code refactoring/restructuring,
  - embedded version control with Git
  - install packages as extensions using VS Code Marketplace
- Lets users use the interface to edit & run their local code/jupyter/R directly on the cluster without having to use OOD, `sbatch`, or `salloc`

Install & Launch VS Code

- OS-based installation:
  - Download: https://code.visualstudio.com/download
  - Linux: Running Visual Studio Code on Linux
  - macOS: Running Visual Studio Code on macOS
  - Windows: Running Visual Studio Code on Windows

- Launch locally - macOS example
  - Terminal: `code &`
  - Applications -> VS Code icon
  - Command+Spacebar -> Code

VS Code Remote Development

1. VSCode Remote Development via SSH or Tunnel – FASRC DOCS
   - Remote Tunnel
     - Interactive & Sbatch (FASRC recommended)
   - Remote SSH
     - Interactive & needs SSH config file
   - Prerequisites satisfied

2. Open OnDemand (OOD/VDI) Remote Desktop: How to open software – FASRC DOCS
   - Remote development work & seamless integration not required
   - Resilient to network glitches

Pros & Cons

Approach	Type	Pros	Cons
Remote Tunnel via sbatch	Batch job submission	1. Resilient to network glitches 2. Session launches on compute node 3. Only method to launch session on FASSE compute nodes [**caution**: personal systems must be configured according to Minimum PrivSec Responsibilities] 4. Supports launching session on Windows	1. Multi-step process to launch session 2. Session cannot be launched directly from personal device. Login to cluster to submit batch job 3. Allows for single VSCode session only, cannot run concurrent sessions 4. Edit batch file for compute node resource allocation
Remote Tunnel interactive	Interactive job	1. Supports launching sessions on both login & compute nodes 2. Same as #4 above	1. Not resilient to network glitches 2. Same as #1, 2, & 3 above
Remote SSH - ProxyCommand	Interactive job	1. Least steps involved to launch a session 2. Launched directly from personal device 3. Works for both login & compute nodes 4. Supports tandem sessions on CPU & GPU nodes (but not on nodes of the same type), along with a Tunnel session	1. Edit to SSH config file for compute node resource allocation 2. Not resilient to network glitches 3. Does not support compute node session on Windows 4. Needs multiple tries for gpu node
Remote SSH - ProxyJump	Interactive job	Same as #2 & 3 above [#4 not tested]	1. Multi-step process for compute node 2. Same as #2 & 3 above

Tunnel - Sbatch (FASRC Recommended)

- FASRC Recommended
- Remote - Tunnel: Sbatch
  - Resilient to network disruptions
  - Launches the tunnel as sbatch job - vscode.job
  - *sbatch vscode.job*
  - *scancel <JOBID>*

```
#!/bin/bash
#SBATCH -p test          # partition. Remember to change to a desired partition
#SBATCH --mem=4g         # memory in GB
#SBATCH --time=04:00:00 # time in HH:MM:SS
#SBATCH -c 1             # number of cores

set -o errexit -o nounset -o pipefail
MY_SCRATCH=$(TMPDIR=/scratch mktemp -d)
echo $MY_SCRATCH

#Obtain the tarball and untar it in $MY_SCRATCH location to obtain the
#executable, code, and run it using the provider of your choice
curl -Lk 'https://code.visualstudio.com/sha/download?build=stable&os=cli-alpine-x64' | t

#VSCODE_CLI_DISABLE_KEYCHAIN_ENCRYPT=1 $MY_SCRATCH/code tunnel user login --provider git
VSCODE_CLI_DISABLE_KEYCHAIN_ENCRYPT=1 $MY_SCRATCH/code tunnel user login --provider micr

#Accept the license terms & launch the tunnel
$MY_SCRATCH/code tunnel --accept-server-license-terms --name cannontunnel
```

Remote - Tunnel - Interactive

- Remote - Tunnel: Interactive

1. Multi-step process compared to Remote - SSH
2. Needs a tarball to create an executable, *code*, on the cluster
3. Interactive:
   - Add *code* to your path using *~/.bashrc*
   - Go to a compute node & execute: *code tunnel*
   - Follow instructions to launch tunnel using either Github or Microsoft
   - Open a browser & authenticate
   - Will have to follow the process every time for a new compute node

Remote - SSH

- SSH Config File:
  - Access login node via SSH control master
  - Generate SSH public & private key pair for compute node
  - Access compute node using ProxyCommand & salloc
  - Or access compute node using ProxyJump

- ProxyCommand+salloc: Either edit local SSH config file or create multiple hostnames. Multiple retries for launching on GPU node
- ProxyJump - Multi-step process to open interface on compute node
- Both are **interactive** - prone to network disruptions

Remote - SSH

ProxyCommand

```
Host cannon
#User mjoshi
User mjoshiunpriv
HostName login.rc.fas.harvard.edu
ControlMaster auto
ControlPath ~/.ssh/%r@%h:%p


Host vscode
    UserKnownHostsFile=/dev/null
    ForwardAgent yes
    StrictHostKeyChecking no
    LogLevel ERROR
    # substitute your username here
    User mjoshi
    #User mjoshiunpriv
    RequestTTY yes
    # Uncomment the command below to get a GPU node on
    #ProxyCommand ssh -q cannon "salloc --immediate=180
    # Uncomment the command below to get a non-GPU node
    ProxyCommand ssh -v cannon "salloc --immediate=180

Host vscode_gpu
    UserKnownHostsFile=/dev/null
    ForwardAgent yes
    StrictHostKeyChecking no
    LogLevel ERROR
    # substitute your username here
    #User mjoshi
    User mjoshiunpriv
    RequestTTY yes
    # Uncomment the command below to get a GPU node on
    ProxyCommand ssh -q cannon "salloc --immediate=180
```

ProxyJump

```
Host cannon
  HostName holylogin01.rc.fas.harvard.edu
  User <username>
  ControlMaster auto
  ControlPath ~/.ssh/%r@%h:%p

Host holy*
    HostName %h
    User <username>
    ProxyJump cannon
```

Best Practices

- Maximum of 5 login sessions allowed per user at a time, be aware of the number of VS Code instances you spawn on the cluster
- Login node session
  - Use for writing &/or editing your code **only**
  - **Do not** use it to run Jupyter notebook, R, Matlab, or any other script
- Compute node session
  - Use for running notebooks & scripts
  - Avoid using for writing &/or editing your code as this is a non-compute work
- For interactive sessions, better to be on VPN to get stable connection
- Close jobs, launched through interactive or sbatch sessions, if VS Code work is complete: `squeue -u <username>; scancel <JOBID>`

Pitfalls

- Lingering SSO connection

```
$ ssh -O check cannon
Master running (pid=#)
$ ssh -O exit cannon
Exit request sent.
$ ssh -O check cannon
Control socket connect(<path-to-connection>): No such file or directory
```

- VSCode running slow, environment issues: *Cache*, *CachedData*, *CachedExtensionsVSIXs*, *Code Cache*, etc.
  - **On Linux:** *.vscode/data/ & .vscode-server/data/* (if opening through remote)
  - **On Mac:** *~/Library/Application\ Support/Code/*
  - **On C:** *C:\Users\<user_name>\AppData\Roaming\Code*

Pitfalls contd…

- **Remote SSH**: Using different nicknames

```
Host cannon
User <username>
HostName login.rc.fas.harvard.edu
ControlMaster auto
ControlPath ~/.ssh/%r@%h:%p
```

```
Host compute
UserKnownHostsFile=/dev/null
ForwardAgent yes
StrictHostKeyChecking no
LogLevel ERROR
# substitute your username here
User <username>
RequestTTY yes
# Uncomment the command below to get
the 2nd ProxyCommand
#ProxyCommand ssh -q cannon "salloc
gres=gpu:1 --time=0-01:00 --mem=4GB -
job-id; nc \$SLURM_NODELIST 22'"
```

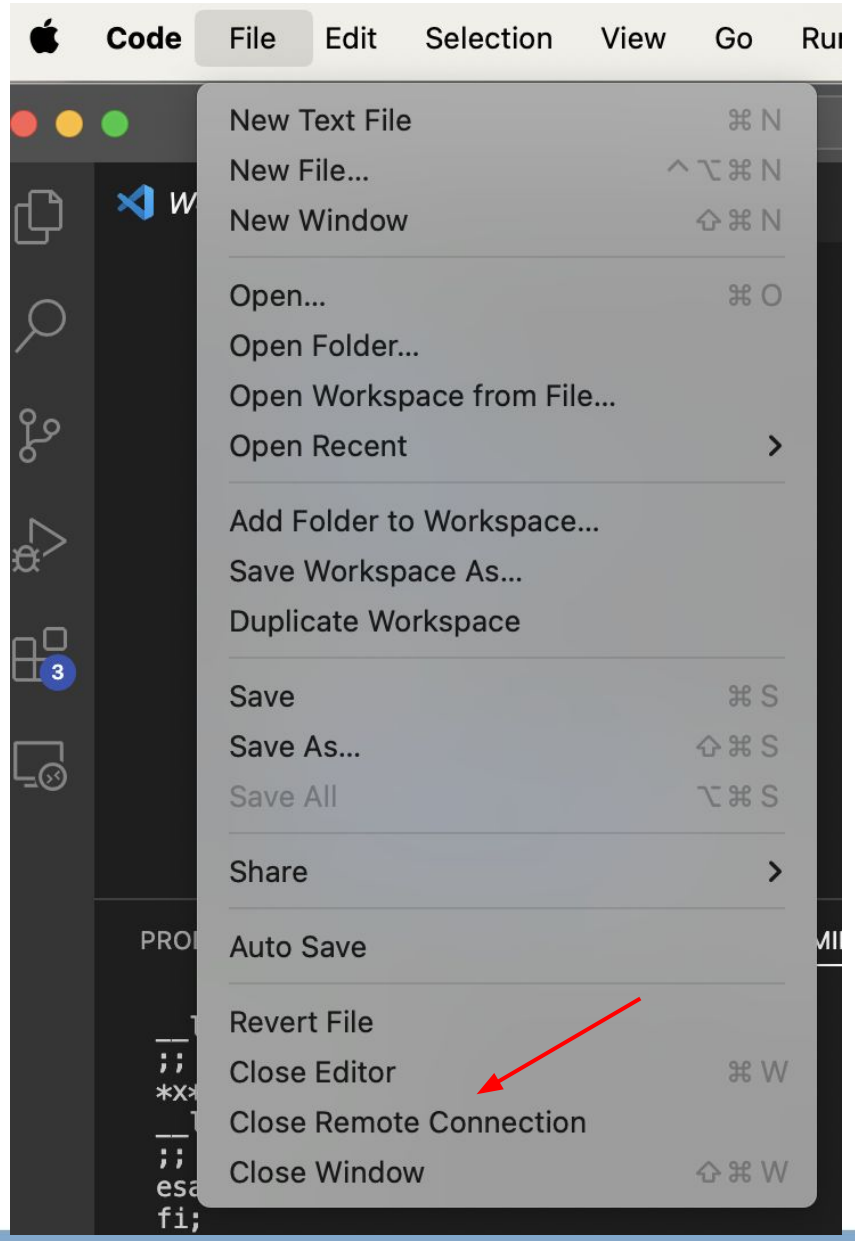
Pitfalls contd…

- **Remote SSH:** Slurm Directives
  - `--mem` flag is **not** being used in `salloc` command
  - Forgot to change `--mem, --time, --partition`, etc., in `salloc` command based on your need
  - Out Of Memory error: forgot to increase memory using `--mem` flag in salloc command prior to launching VSCode session on the cluster
  - SSH config file not setup correctly:
    - Test on macOS: `ssh <Host Nickname>`
      - Replace `<Host Nickname>` with names used for login & compute nodes
- Not commenting out conda initialization statements in *~/.bashrc*
- Connectivity issues: $HOME could be full

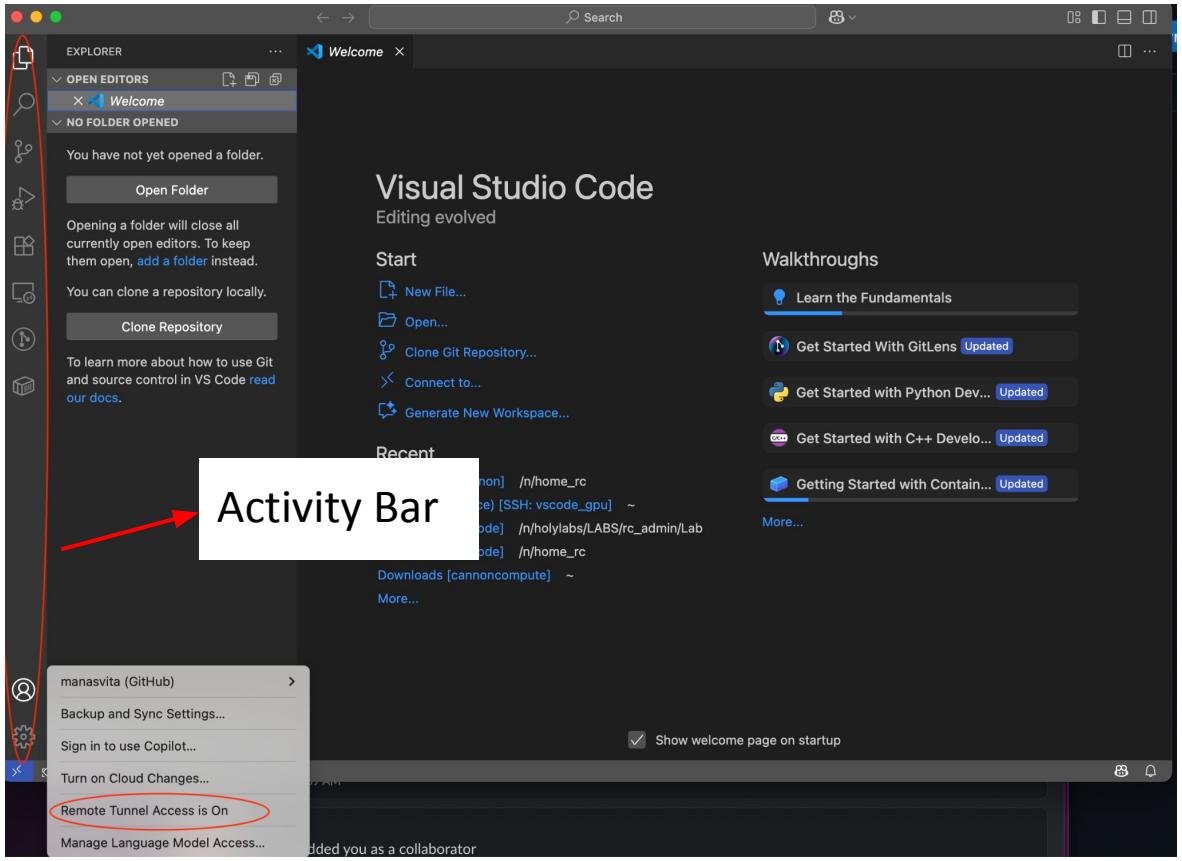
Pitfalls contd…

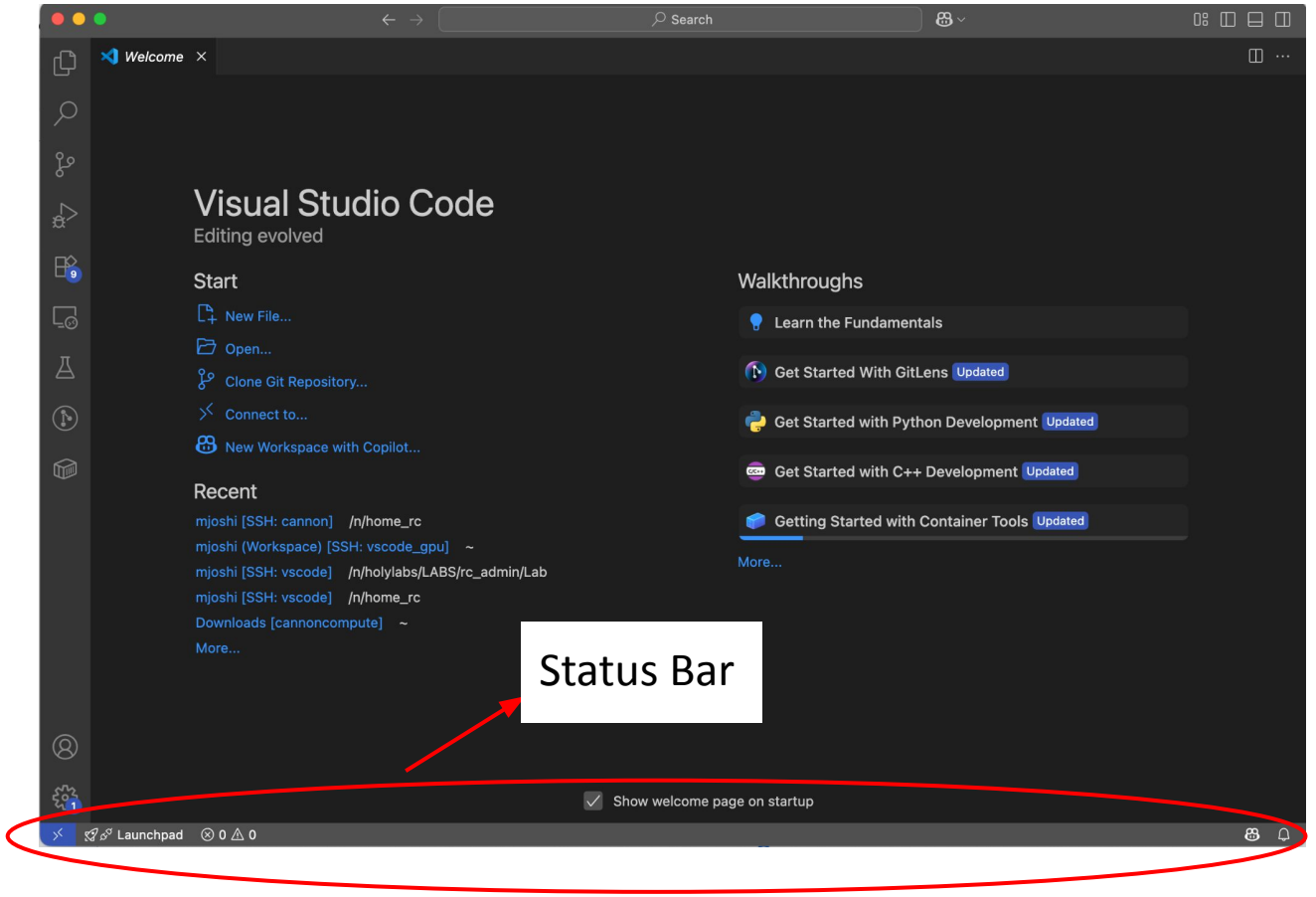
- Not exiting cleanly:
  - Close Remote Connection under File
  - Having multiple windows open
  - Important for Remote - Tunnel connections

- Continue to have problems:
  - Come to office hours to troubleshoot live



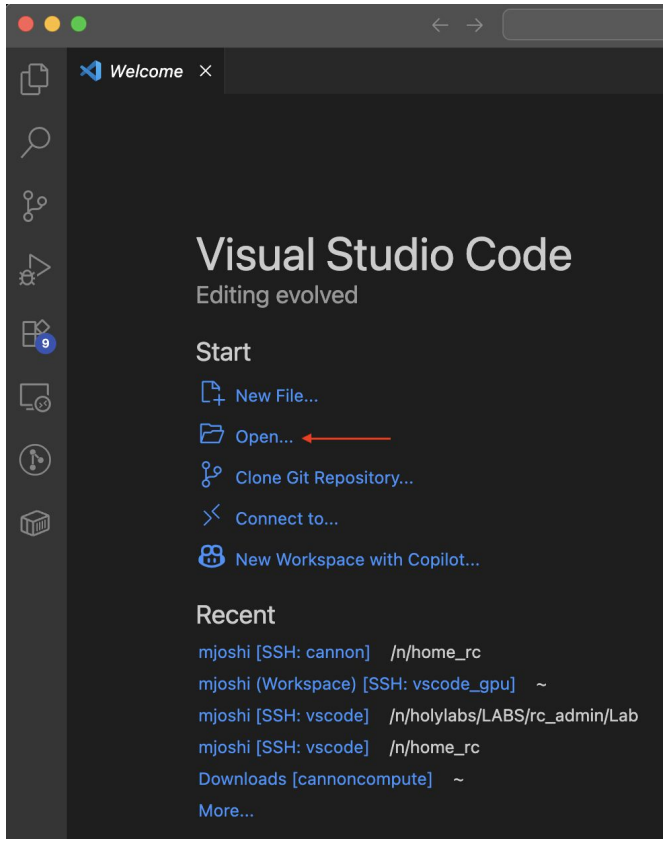
VSCode Basics



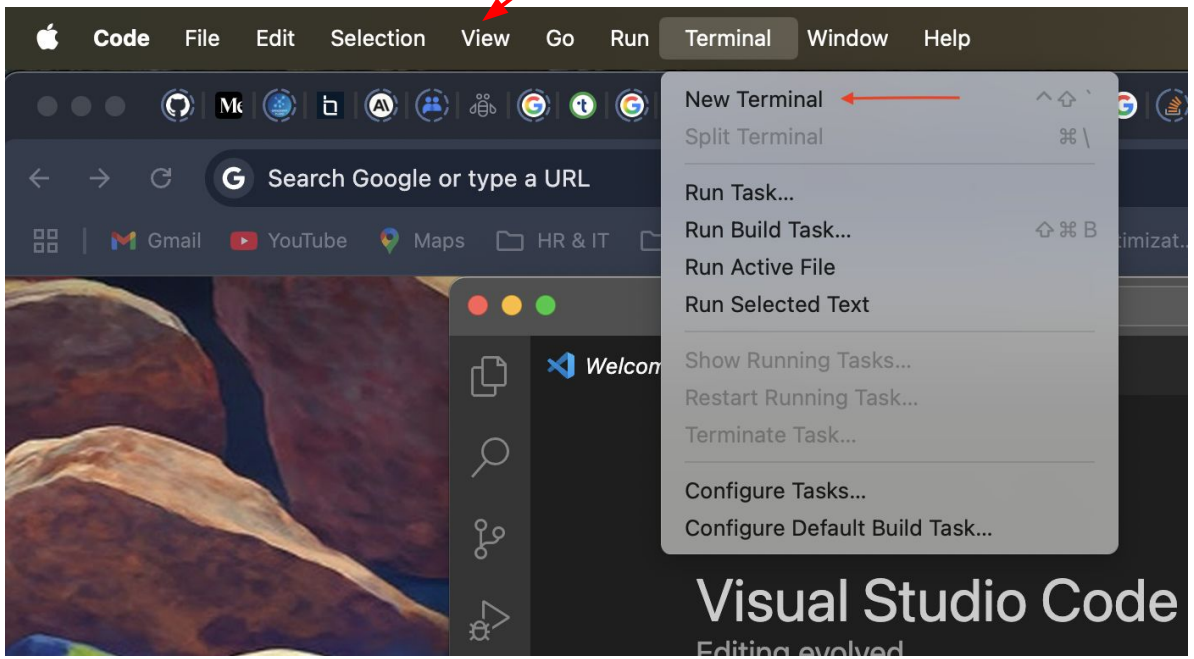


VSCode Basics

- Folder: cmd+o (Mac) or ctrl+o (Win)



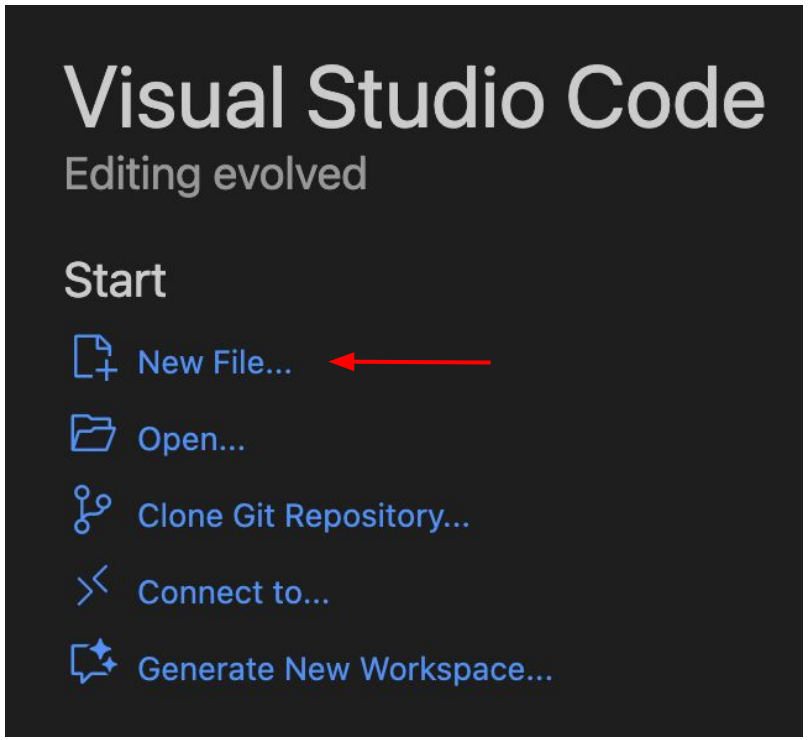
- Terminal
  - View -> Command Palette -> >terminal (select from options)
  - Using top panel



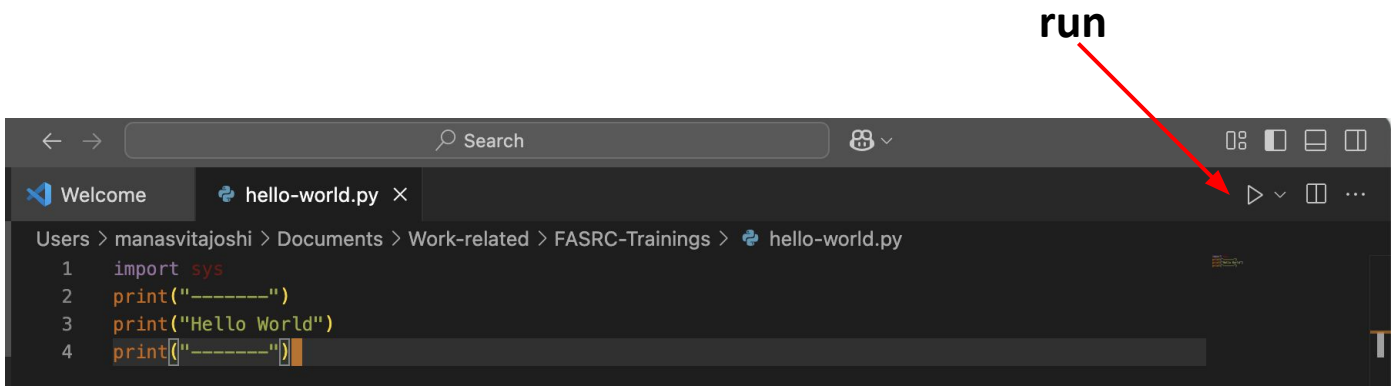
Getting started: https://code.visualstudio.com/docs/introvideos/basics

VSCode Basics

- File: cmd+n (Mac) or ctrl+n (Win)



- Run jobs interactively



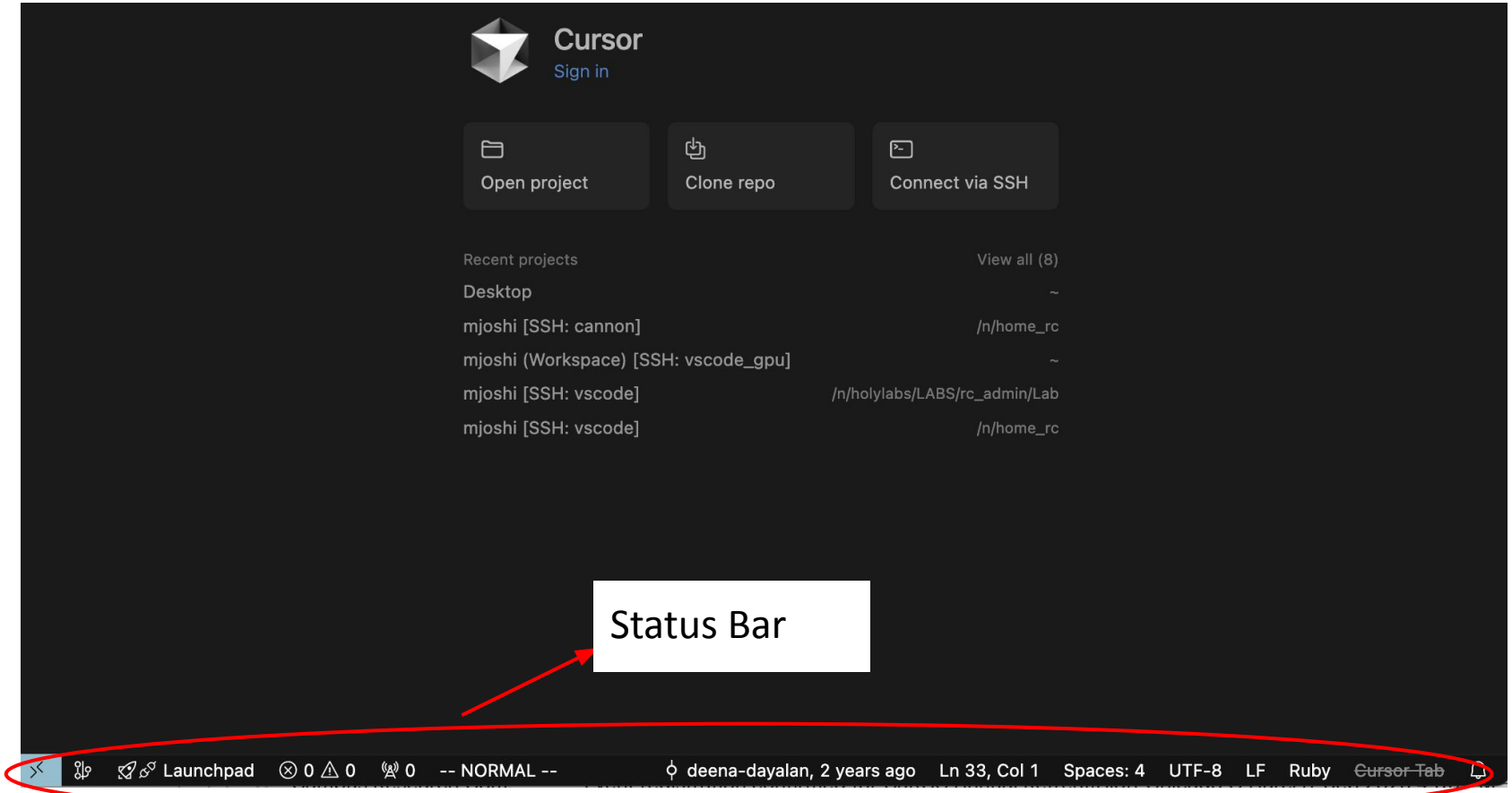
- Once connected, `run` executes code directly on Remote
- ***Always connect to compute node before hitting `run`***

Other Considerations

- VSCode for FASSE:
  - Enable access to the internet to launch Remote Tunnel on the browser: https://docs.rc.fas.harvard.edu/kb/fasse/#Accessing_the_Internet
  - Install extensions as needed.
  - https://code.visualstudio.com/docs/configure/settings-sync may not work
- Add folders to workspace: https://docs.rc.fas.harvard.edu/kb/vscode-remote-development-via-ssh-or-tunnel/#Add_Folders_to_Workspace_on_VSCode_Explorer

A little bit about Cursor

- Similar to VSCode: Launching local instance



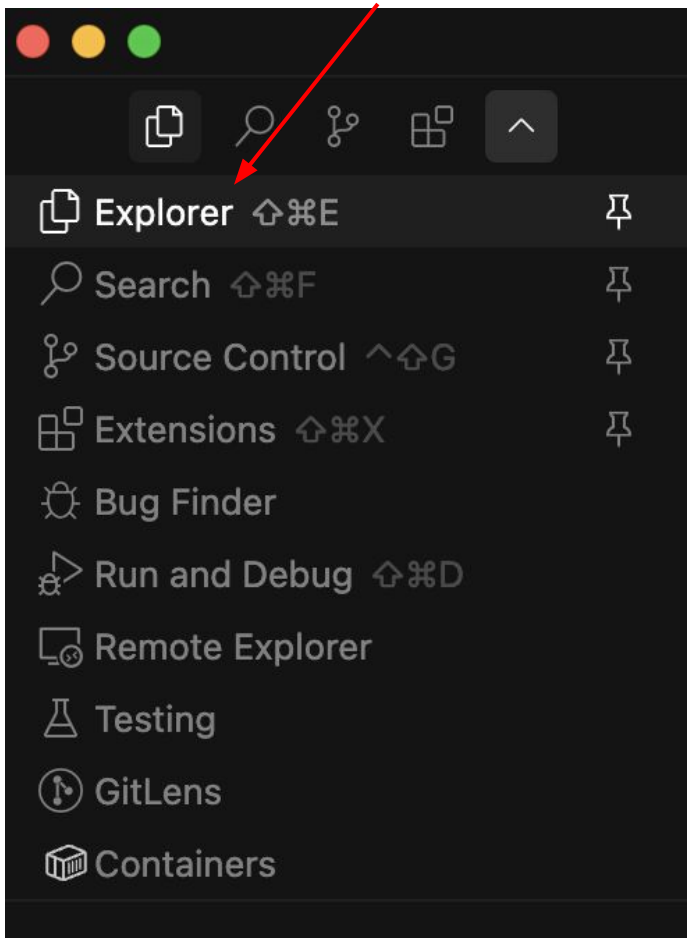
Download for MacOS & other systems: https://cursor.com/

Installation: https://docs.cursor.com/en/get-started/installation

FASRC Doc

A little bit about Cursor

- Remote SSH - Same ~/.ssh/config works
- Remote Tunnel support since June 8, 2024: changelog 0.35: https://changelog.cursor.sh/#035---default-on-cursor-prediction-remote-tunnels--more-robust-ssh
- Both, interactive & sbatch, options supported
- No option for opening tunnel on a browser
- Could have issues, VSCode tunnel might be better: https://forum.cursor.com/t/cursor-vs-code-microsoft-remote-tunneling-issues-and-the-definitive-support-thread/85883



1. curl -Lk 'https://api2.cursor.sh/updates/download-latest?os=cli-alpine-x64' -o cursor_cli.tar.gz
2. tar -xvf cursor_cli.tar.gz
3. ./cursor tunnel --random-name

Credit: https://github.com/cursor/cursor/issues/1191#issuecomment-3139267445

Resources:

- VSCode Remote Development via SSH or Tunnel – FASRC DOCS
- Kempner's VSCode Remote Development: https://handbook.eng.kempnerinstitute.harvard.edu/s1_high_performance_computing/development_and_runtime_envs/using_vscode_for_remote_development.html
- SSH wrapper script for launching VSCode as a background job: pretty exhaustive:
  - https://github.com/microsoft/vscode-remote-release/issues/1722
  - https://github.com/xangma/vscode_remote_slurm/tree/main
- Documentation for Visual Studio Code

Resources and help

- Documentation
  - User Docs: FASRC DOCS
  - GitHub User codes: GitHub - fasrc/User_Codes
- Getting help
  - Office hours on Wednesdays from 12-3 PM: https://www.rc.fas.harvard.edu/training/office-hours/
  - Ticket
    - Portal: http://portal.rc.fas.harvard.edu/rcrt/submit_ticket (requires login)
    - Email: rchelp@rc.fas.harvard.edu
  - Consulting Calendar: https://www.rc.fas.harvard.edu/consulting-calendar/
- Training: Training | FAS Research Computing

Training Session Evaluation

Please, fill out our training session evaluation. Your feedback is essential for us to improve our trainings!!

https://tinyurl.com/FASRC-training

Thank you :)

FAS Research Computing

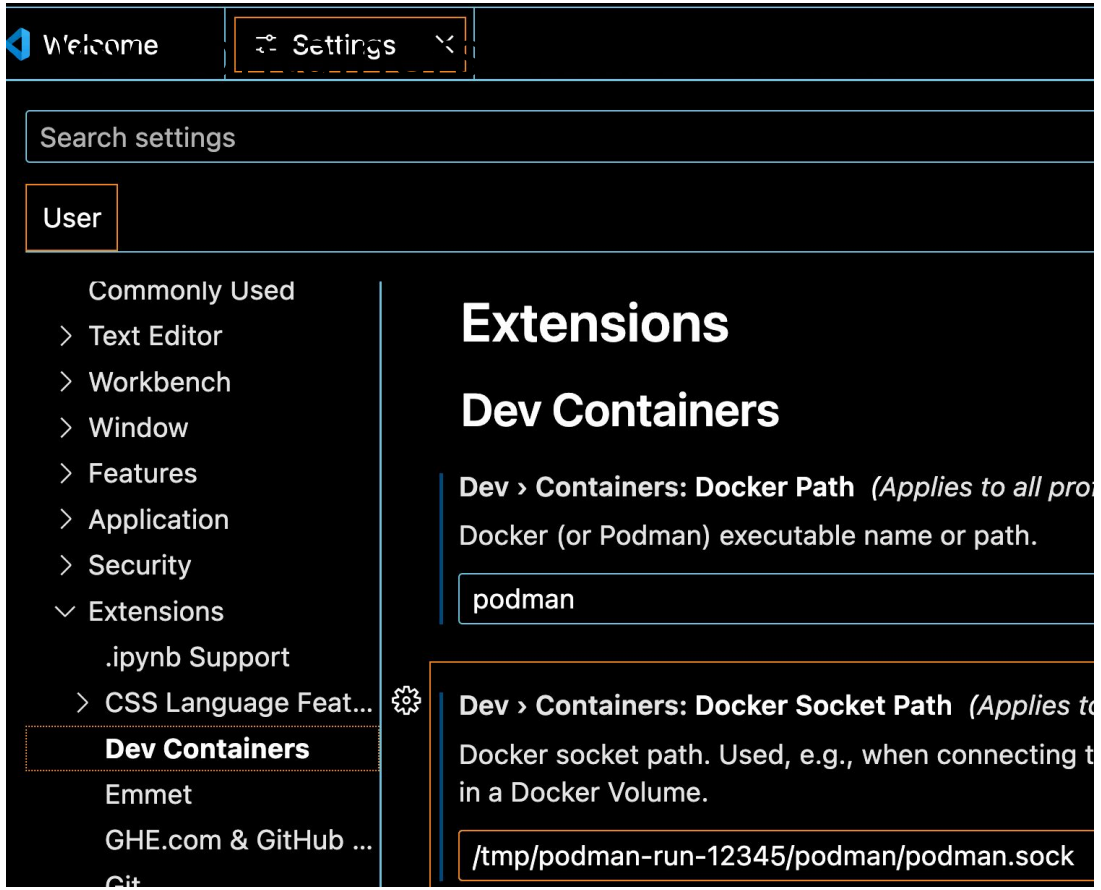
Dev Containers with Podman

- VS Code Dev Containers extension "lets you use a Docker container as a full-featured development environment"
- Repository contains a devcontainer.json file
  - defined by Dev Container spec (https://containers.dev/)
  - example: nextflow-training (local-dev)

To use on the FASRC cluster:

1. Install VS Code Dev Container extension
2. Configure to use Podman:
   a. Code > Settings > Settings > Extensions > Dev Containers
      i. Change "Docker Path" setting to "podman"
      ii. Set Docker Socket path to:

`/tmp/podman-run-<uid>/podman/podman.sock`
(Replace <uid> with number output by "id -u")



Dev Containers with Podman - Known Limitations

- Not all dev container features are supported (e.g., **Docker***)
- Not supported by vscode.dev (issue #9059) – Remote Tunnel users must use VS Code app instead of web browser
- Rebuilding a dev container (after modifying devcontatiner.json) fails
  - possibly containers/conmon issue #260
  - workaround: close SSH session (or scancel <jobid> for remote tunnel) and start new session