

# AI/ML for prediction of biological properties of molecules

Module 2. Setting up the computational environment

Gemma Turon & Miquel Duran-Frigola  
Ersilia Open Source Initiative ([www.ersilia.io](http://www.ersilia.io))  
18th - 27th of September, 2023



# Let's get started!

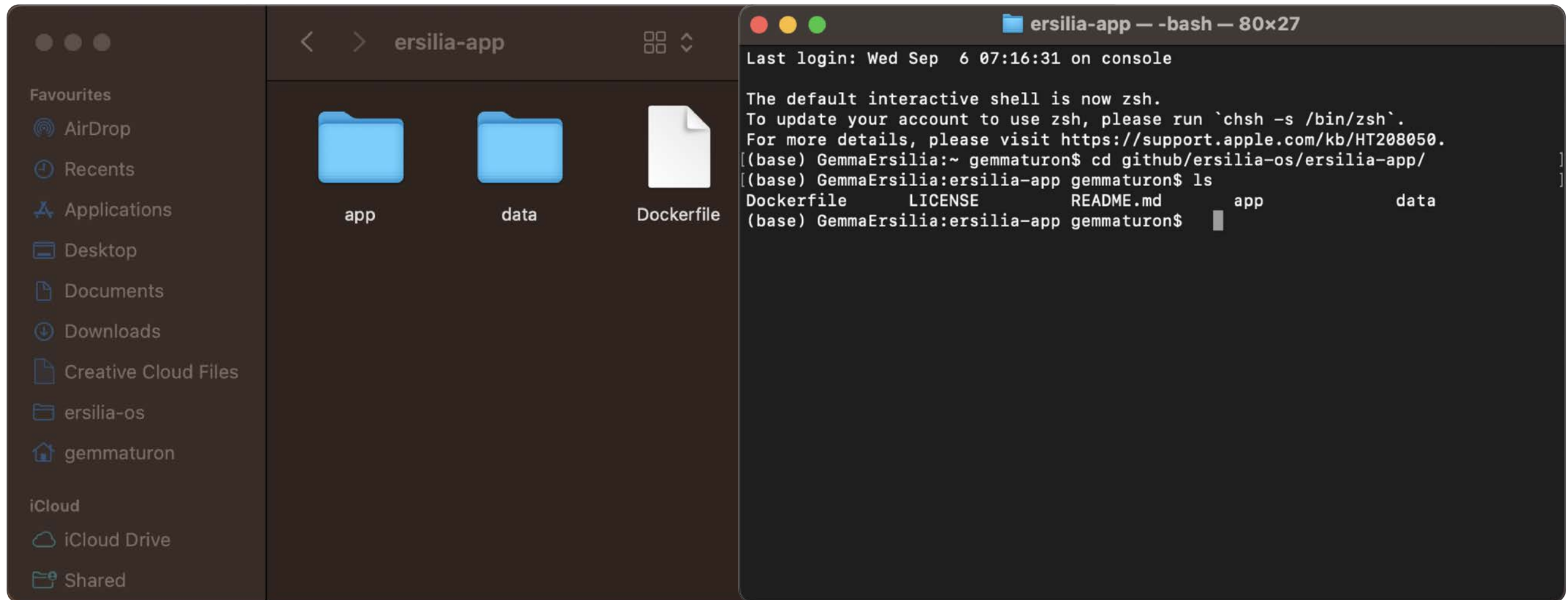
Go to [menti.org](https://menti.org) and introduce  
this code: 123456789





# The Command Line Interface (CLI)

# Command Line Interface (CLI) vs Graphical User Interface (GUI)



The CLI sends commands in the form of lines of text to the computer, as opposed to the GUI, which has clickable menus

# Find the CLI in your system

The programming language of the CLI or Terminal varies between systems

- Windows: ~~command prompt (CMD)~~
- UNIX (MacOS and Linux): bash and zsh

👉 Windows users: download and install PowerShell 7.0

👉 UNIX users: open the Terminal

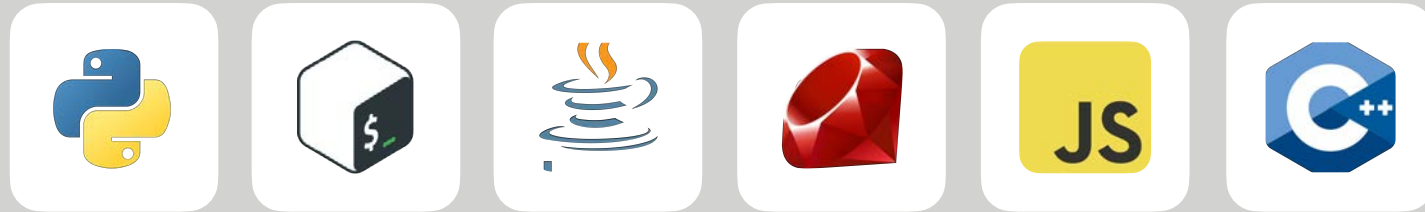
💡 Use the search function if you can't locate the Terminal

# Uses of the CLI

- Automate repetitive tasks
- Pipe several commands, eliminating the need for generating intermediate files and steps
- Use programs that do not offer a GUI:
  - Send API queries to download data from an online repository
  - Access and operate remote computers
- ...

# Python and Package managers

# Multiple programming languages



Programming languages are sets of rules that convert human-readable text to machine code

Their instructions are precise and only have one single meaning



- Human readable syntax
- Flexibility for several applications
- Lots of pre-built packages



# Python packages

```
# Install package via manager (pip or conda)

!pip install rdkit

# Import modules or functions from the package

import rdkit
from rdkit import Chem

# Call a specific function to use it
molecule =
Chem.MolFromSmiles("CCCCOCCCC")
```

Python packages are collections of reusable Python modules that provide pre-written code and functionality. They simplify code reuse and maintenance.

# Types of Python files: scripts

```
# Import modules or functions from the package

import rdkit
from rdkit import Chem

# Call a specific function to use it
molecule = Chem.MolFromSmiles("CCCCOCCCC")
inchikey = Chem.inchi.MolToInchiKey(molecule)

# Print the output of the functions
print(inchikey)
```

```
# Run a python script
```

```
(mycondaenv) $ python myscript.py
DURPTKYDGMDSBL-UHFFFAOYSA-N
```

- A script is a .py file
- It has a series of commands that are run one after the other
- At the top, we need to import the required packages

- We run it via the CLI

# Types of python files:

## Jupyter notebooks

A notebook allows us to run small pieces of code at a time, making it very easy to perform data analysis.

The screenshot shows a Jupyter Notebook environment with a dark theme. The Explorer sidebar on the left lists files: `cli.sh`, `mynotebook.ipynb`, and `python.py`. The main area displays two code cells. The first cell imports `os`, `pandas` as `pd`, and `rdkit.Chem`, and sets `DATAPATH = "../data"`. The second cell uses `pd.read_csv` to load `chembl_50k.csv` into a DataFrame `df`. Below the code, the DataFrame is displayed as a table with 50,000 rows and 1 column named `smiles`. The table shows a preview of the data with rows indexed from 0 to 49,999. The bottom status bar indicates the environment is `chem (Python 3.10.9)`.

```
import os
import pandas as pd
from rdkit import Chem

DATAPATH = "../data"
```

```
df = pd.read_csv(os.path.join(DATAPATH, "chembl_50k.csv"))
df
```

	smiles
0	<chem>N#CC1CCCN(C(=O)CCC2=CC=CC(F)=C2)C1</chem>
1	<chem>CCC(=O)C1=CN=C2C=CC(C3=CC(Cl)=C(O)C(OC)=C3)=CC...</chem>
2	<chem>C/C(=N\NC(=O)C1=NC2=C(C(=O)N1)C1CCCN1C(=O)N2C1...</chem>
3	<chem>CC(=O)N[C@H]1[C@H](SCCCN2C=C(CN3C(=O)C4=CC=CC5...</chem>
4	<chem>OC1=C(/C=N/C2=NC=CS2)C2=CC=CC=C2N1</chem>
...	...
49995	<chem>N=C(N)N/N=C/C1=CC=C(Cl)C=C1OCC1=CC=C(Cl)C=C1</chem>
49996	<chem>NNC(=O)CC1=NNC(O)=C1</chem>
49997	<chem>O=C1[C@H](NS(=O)(=O)C2=CC=C(OC3=CC=C(Cl)C=C3)C...</chem>
49998	<chem>CC(C)(CCCC1=CC=C(CCCC(C)(C)CC(=O)O)C=C1)CC(=O)O</chem>
49999	<chem>O=C(CCN1CCN(S(=O)(=O)C2=CC=C(F)C=C2)CC1)NC1=CC...</chem>

50000 rows x 1 columns

# Where does Python run?

Operating system (Windows, Linux, MacOS)

Conda env  
for project  
1

- Python 3.9
- RdKit
- Matplotlib
- Seaborn

Conda env  
for project  
2

- Python 3.10
- Scikit-Learn
- FLAML
- ChemicalChecker

Conda env  
for project  
3

- Python 3.10
- ...

`myscript.py`

```
graph LR; A[Conda env for project 1] --> D[myscript.py]; B[Conda env for project 2] --> D; C[Conda env for project 3] --> D;
```

The diagram illustrates the execution environment for a Python script. It shows three distinct Conda environments, each associated with a specific project. Each environment contains a unique set of dependencies, including different versions of Python and various libraries. Arrows from each environment point towards a central script file named 'myscript.py', indicating that the script can be executed within any of these environments.



# Where does Python run?

Operating system (Windows, Linux, MacOS)

Conda env  
for project  
1

- Python 3.9
- RdKit
- Matplotlib
- Seaborn

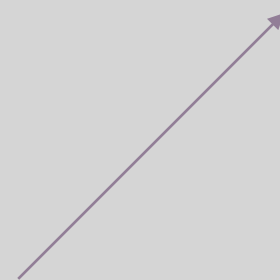
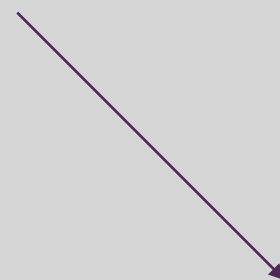
Conda env  
for project  
2

- Python 3.10
- Scikit-Learn
- FLAML
- ChemicalChecker

Conda env  
for project  
3

- Python 3.10
- ...

```
import rdkit
```



# Where does Python run?

Operating system (Windows, Linux, MacOS)

Conda env  
for project  
1

- Python 3.9
- RdKit
- Matplotlib
- Seaborn

Conda env  
for project  
2

- Python 3.10
- Scikit-Learn
- FLAML
- ChemicalChecker

Conda env  
for project  
3

- Python 3.10
- ...

```
import rdkit
```



# How do we run Python?

## Operating system (Windows, Linux, MacOS)

Conda env  
for project  
1

- Python 3.9
- RdKit
- Matplotlib
- Seaborn

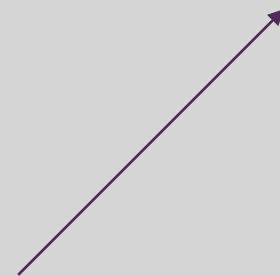
Conda env  
for project  
2

- Python 3.10
- Scikit-Learn
- FLAML
- ChemicalChecker

Conda env  
for project  
3

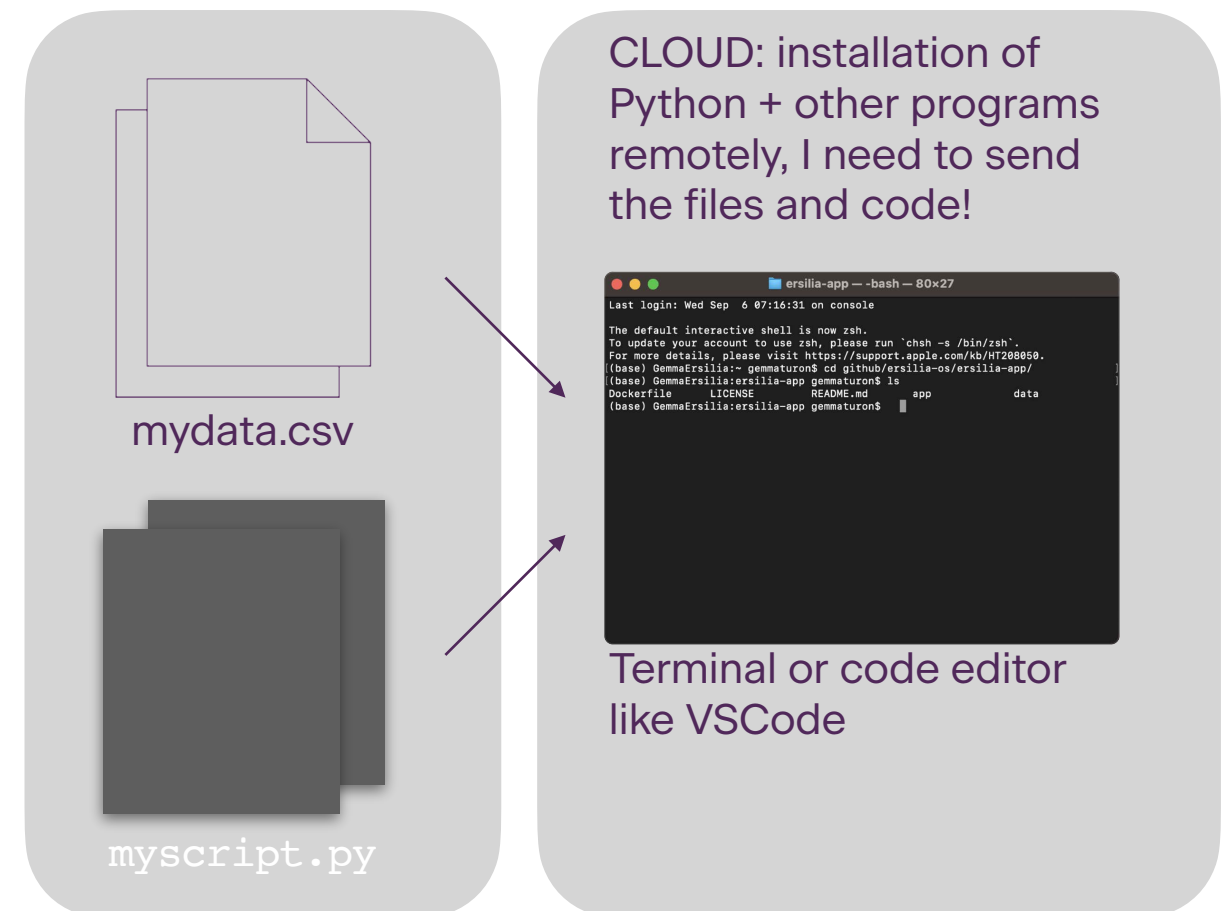
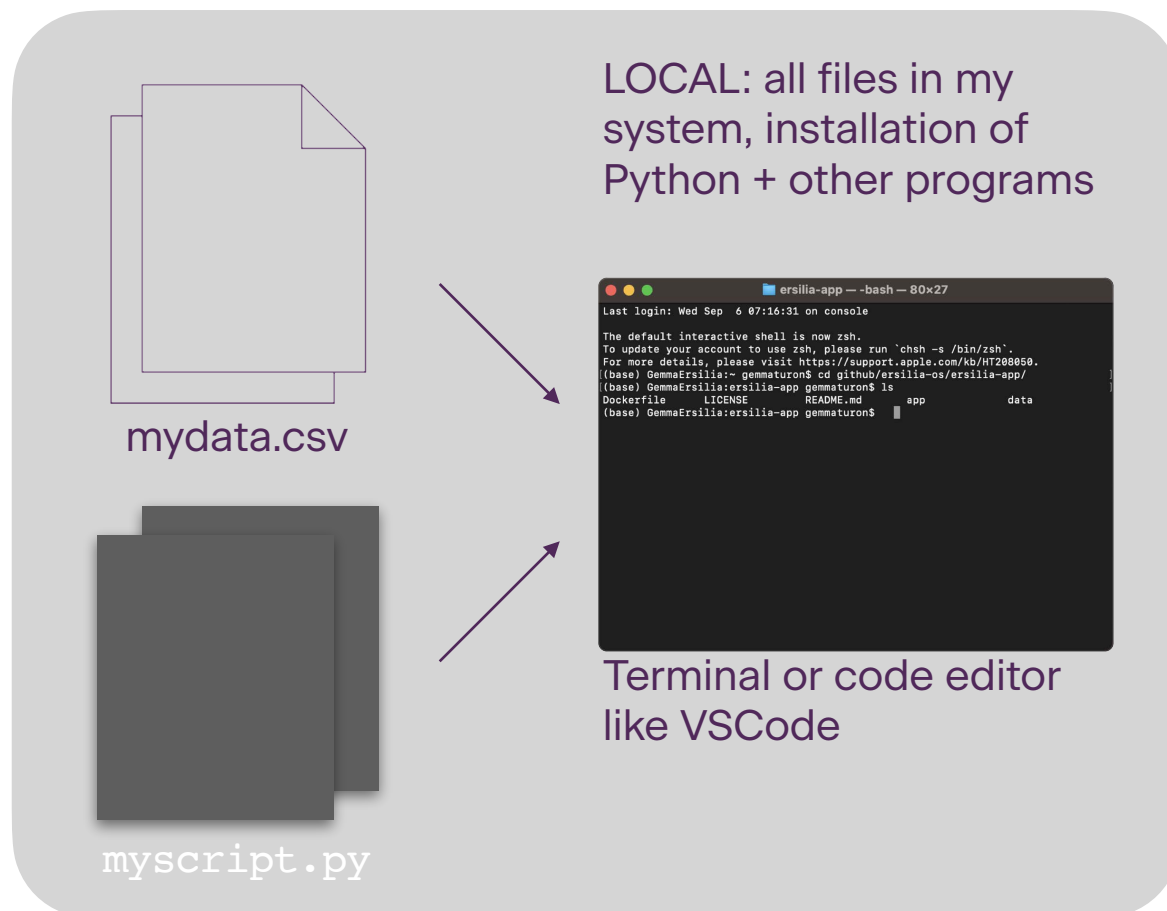
- Python 3.10
- ...

```
import rdkit
```



# Where does Python run?

- Locally: we need python installed as well as a package manager like Anaconda
- Remote computer: all the packages are installed in the remote computer





# Git and GitHub

# What is Git? What is GitHub?



Git is a software that tracks changes to files, including creating new files, deleting files and folders, and editing content



GitHub is an internet hosting service for software development and version control using Git

# The importance of GitHub 🚀

- 100 million developers
- 372 million repositories (28 million are public)
- More than 20,000 academic papers are referenced in GitHub repositories

The screenshot shows the GitHub profile page for the 'Ersilia Open Source Initiative'. The header includes the repository name 'ersilia-os', a search bar, and navigation links for Overview, Repositories (229), Projects (2), Packages, Teams (3), People (16), and Settings. The profile section features a green repository icon, the name 'Ersilia Open Source Initiative', a description: 'Ersilia is a charity developing open source tools to facilitate global health drug discovery, with a focus on neglected diseases, for equal healthcare', and contact information: 110 followers, United Kingdom, website http://ersilia.io, and social media handles @ersiliaio and hello@ersilia.io. An 'Unfollow' button is present. Below the profile, the 'README.md' file is displayed, starting with a green virus icon and the text 'Welcome to Ersilia 🧬'. The README lists four bullet points: 1. The Ersilia Open Source Initiative is a non-profit organization dedicated to promoting open-source science and fostering a collaborative community. 2. Our mission is to equip laboratories in the Global South with machine learning and data science tools for drug discovery, empowering them to address local and global health challenges. 3. Browse the Ersilia Model Hub, our repository of pre-trained machine learning models! 4. Visit more extensive documentation in the Ersilia Book. On the right side of the README, there is a 'View as: Public' dropdown menu, a note stating 'You are viewing the README and pinned repositories as a public user.', a link to 'Get started with tasks' that most successful organizations complete, and a 'Discussions' section with the text 'Set up discussions to engage with your community!'.

ersilia-os

Search Type to search

Overview Repositories 229 Projects 2 Packages Teams 3 People 16 Settings

**Ersilia Open Source Initiative**

Ersilia is a charity developing open source tools to facilitate global health drug discovery, with a focus on neglected diseases, for equal healthcare

110 followers United Kingdom http://ersilia.io @ersiliaio hello@ersilia.io

Unfollow

README.md

**Welcome to Ersilia 🧬**

- The [Ersilia Open Source Initiative](#) is a **non-profit organization** dedicated to promoting open-source science and fostering a collaborative community.
- Our mission is to [equip laboratories in the Global South](#) with machine learning and data science tools for drug discovery, empowering them to address local and global health challenges.
- Browse the [Ersilia Model Hub](#), our repository of **pre-trained machine learning models**!
- Visit more extensive **documentation** in the [Ersilia Book](#).

View as: **Public**

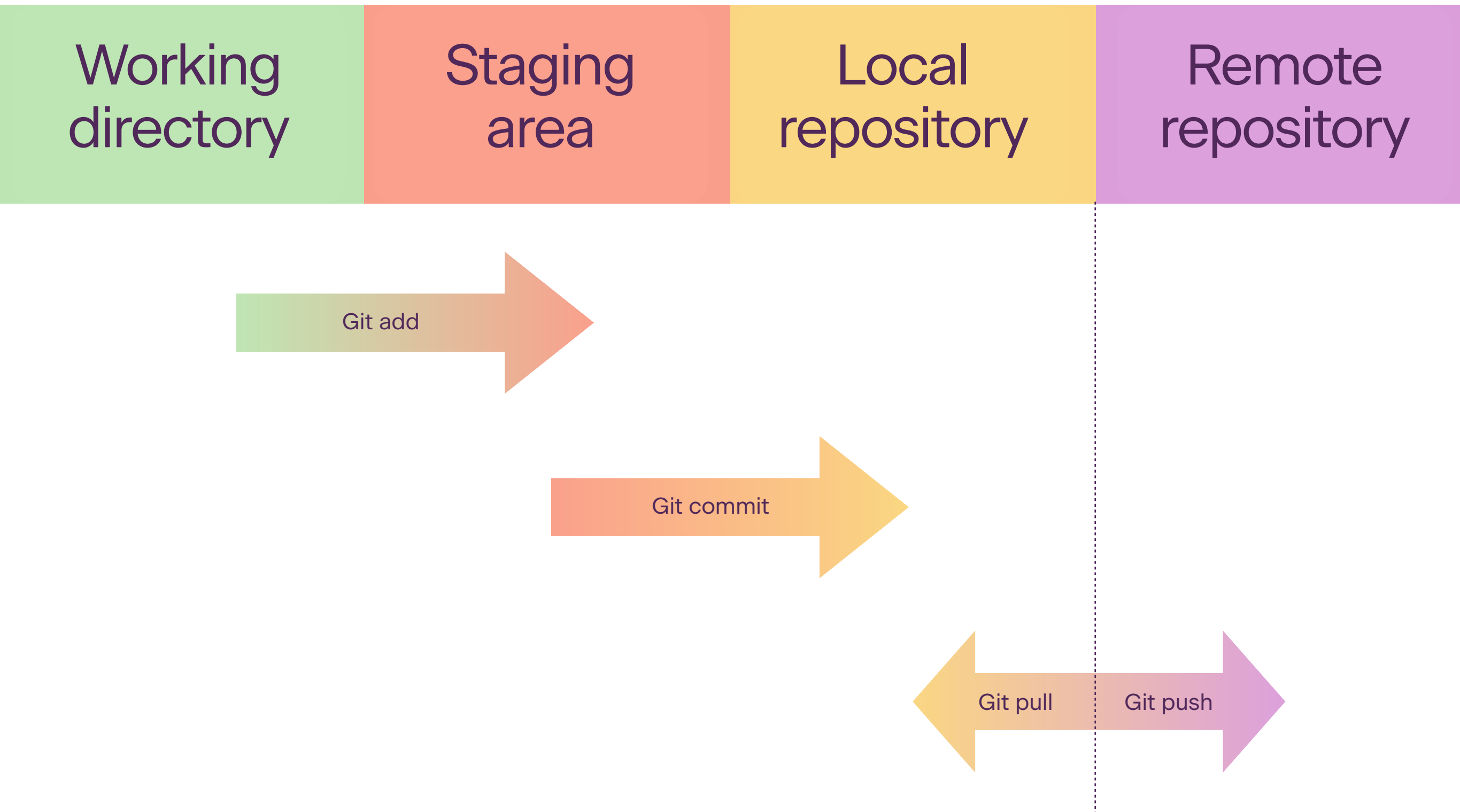
You are viewing the README and pinned repositories as a public user.

[Get started with tasks](#) that most successful organizations complete.

**Discussions**

Set up discussions to engage with your community!

# How does Git work?





Join GitHub! 🤗

<https://github.com/join>

Product ▾ Solutions ▾ Open Source ▾ Pricing

🔍 Search or jump to...

/

Sig

Join GitHub

First, let's create your user account

Username \*

Email address \*

Password \*

Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more.](#)

Email preferences

☐ Send me occasional product updates, announcements, and offers.

Verify your account

Please solve this puzzle so we know you are a real person

# It's like social media - Follow Ersilia!

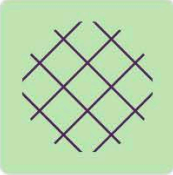
<https://github.com/ersilia-os/>

The screenshot shows the GitHub profile page for the Ersilia Open Source Initiative. The page layout includes a header with navigation links (Overview, Repositories, Projects, Packages, Teams, People, Settings) and a search bar. The main content area features the organization's profile, including its name, description, and contact information. A circular callout highlights the 'Unfollow' button. The README section is visible, containing a welcome message and a list of bullet points about the organization's mission and resources. The right sidebar shows options to view the repository as public or private, a link to get started with tasks, a section for discussions, and a list of people associated with the repository.

github.com/ersilia-os

ersilia-os

Overview Repositories 229 Projects 2 Packages Teams 3 People 16 Settings

 **Ersilia Open Source Initiative**

Ersilia is a charity developing open source tools to facilitate global health drug discovery, with a focus on neglected diseases, for equal healthcare

110 followers United Kingdom <http://ersilia.io> [@ersiliaio](https://twitter.com/ersiliaio) [hello@ersilia.io](mailto:hello@ersilia.io)

Unfollow

README.md

## Welcome to Ersilia 🧬💊

- 🏠 The [Ersilia Open Source Initiative](#) is a **non-profit organization** dedicated to promoting open-source science and fostering a collaborative community.
- 🌍 Our mission is to **equip laboratories in the Global South** with machine learning and data science tools for drug discovery, empowering them to address local and global health challenges.
- 👤 Browse the [Ersilia Model Hub](#), our repository of **pre-trained machine learning models**!
- 📖 Visit more extensive **documentation** in the [Ersilia Book](#).

## 🌟 Get Involved

We encourage participation from researchers, developers, and enthusiasts around the world. By contributing to our open-source projects, you can help us make a difference in the lives of people in the Global South.

- 🤝 **Collaborate** with us on our projects by contributing code, ideas, or expertise.
- 💡 **Share** our work with your network to increase awareness and support.
- 🚀 **Adopt** our open-source tools and models in your research or organization to help drive scientific progress.

Together, we can make a lasting impact through open-source science and empower the Global South to tackle health challenges with the power of machine learning and data science.

View as: **Public** ▼

You are viewing the README and pinned repositories as a public user.

[Get started with tasks](#) that most successful organizations complete.

### Discussions

Set up discussions to engage with your community!

[Turn on discussions](#)

### People

A grid of 16 circular profile pictures of community members is displayed.

# It's like social media - Track your progress!

GemmaTuron

Q

Type to search

>

+


Overview

Repositories4

Projects

Packages

Stars5



gemmaturon

GemmaTuron · she/her

@ersilia-os co-founder and CEO

Edit profile

51 followers · 4 following

@ersilia-os

gemma@ersilia.io

https://ersilia.io

@TuronGemma

in/gemma-turon

Popular repositories

Customize your pins

ResourcesPublic

Forked from KairoiAI/Resources

Resources made available under CC-BY 4.0 by Kairoi

test-repoPublic

2,631 contributions in the last year

Contribution settings

SepOctNovDecJanFebMarAprMayJunJulAugSep

Mon

Wed

Fri

Learn how we count contributions

LessMore

Contribution activity

2023

2022

2021

2020

Sep 7

September 2023

Created 33 commits in 7 repositories

Created 1 repository

GemmaTuron/test-repo

# Let's recap!

Go to: [menti.com](https://menti.com)

Code: 1271 8086



---

# Extra information

# Basic commands with the CLI

```
#change directory (cd) command
$ cd Desktop
~/Desktop $

#bash is case sensitive
$ cd desktop
bash: cd: desktop: No such file or directory

#make directory (mkdir)
~/Desktop $ mkdir test
~/Desktop $ cd test
~/Desktop/test $

#go back one directory
~/Desktop/test $ cd ..
~/Desktop $
```

\$ `cd` move to a directory

\$ `mkdir` make a new directory

\$ `cd ..` move one directory up

# Basic commands with the CLI

```
# list files and directories (ls)
~/Desktop $ ls
test
folder1
file1

~/Desktop $ cd test
~/Desktop/test $ ls

#create a file
~/Desktop/test $ touch example.txt
~/Desktop/test $ ls
example.txt

#add a line of text
~/Desktop/test $ echo "my first line" >
example.txt
~/Desktop/test $ vim example.txt
```

\$cd move to a directory

\$mkdir make a new directory

\$cd .. move one directory up

\$ls list files in directory

\$touch make a new file

\$echo add content to file

# Basic commands with the CLI

```
# move the files to new directory
~/Desktop/test $ mkdir subfolder
~/Desktop/test $ ls
subfolder
example.txt
~/Desktop/test $ mv example.txt subfolder
~/Desktop/test $ cd subfolder
~/Desktop/test/subfolder $ ls
example.txt

#copy the file
~/Desktop/test/subfolder $ cp example.txt ../
~/Desktop/test/subfolder $ ls
~/Desktop/test/subfolder $

~/Desktop/test/subfolder $ cd ..
~/Desktop/test $ ls
subfolder
example.txt
```

\$cd move to a directory  
\$mkdir make a new directory  
\$cd .. move one directory up  
\$ls list files in directory  
\$touch make a new file  
\$echo add content to file  
\$mv move a file  
\$cp copy a file

# Basic commands with the CLI

```
# eliminate the files we have created
~/Desktop/test $ ls
subfolder
example.txt
~/Desktop/test $ rm example.txt
~/Desktop/test $ ls
subfolder

# use the -r flag to eliminate directories
~/Desktop/test $ rm -r
~/Desktop/test $ ls
~/Desktop/test $

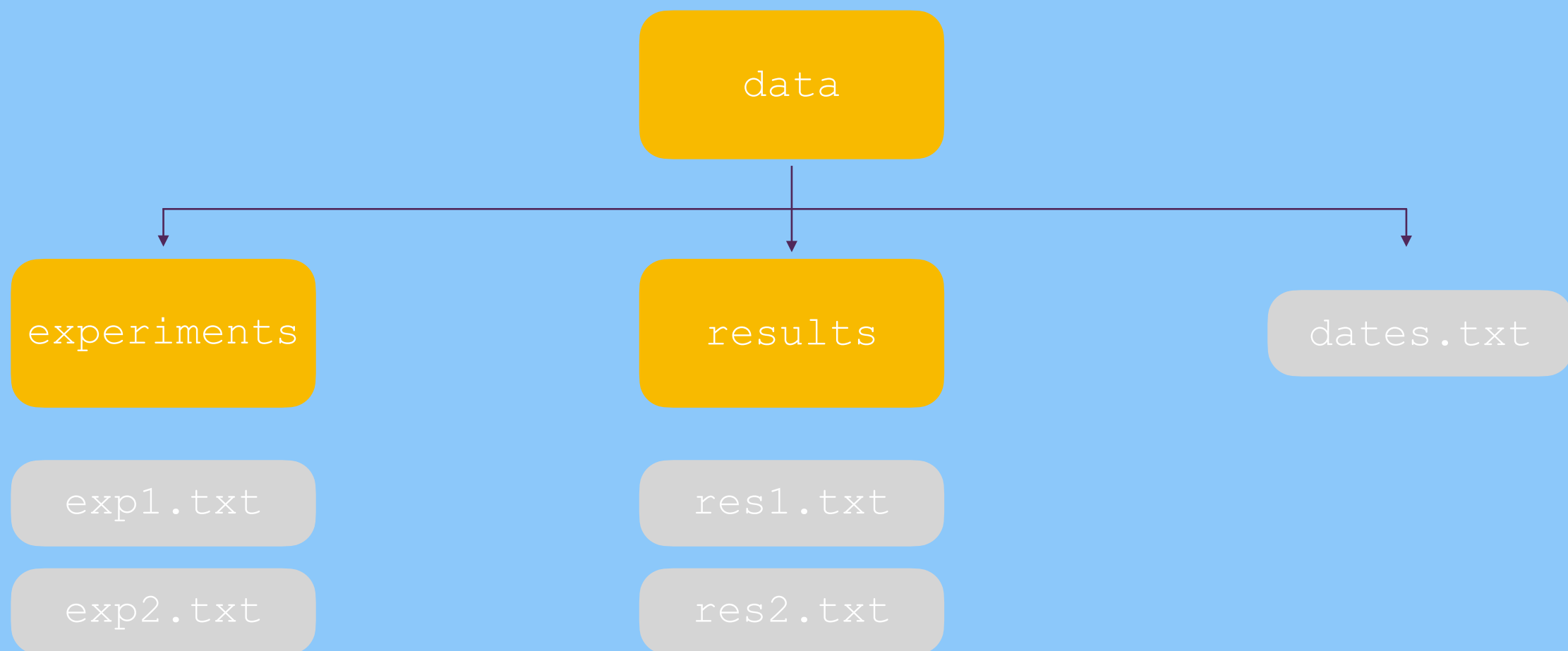
~/Desktop/test $ cd ..
~/Desktop/ $ rm -r test
~/Desktop $ ls
folder1
file1
```

`$ cd` move to a directory  
`$ mkdir` make a new directory  
`$ cd ..` move one directory up  
`$ ls` list files in directory  
`$ touch` make a new file  
`$ echo` add content to file  
`$ mv` move a file  
`$ cp` copy a file  
`$ rm` eliminate file  
`-r` recursive



# Practise CLI commands!

- Using the Command Line Interface create the following directory structure
- Add one line of text to each file





# Git and version control

# What is Git? What is GitHub?



Git is a software that tracks changes to files, including creating new files, deleting files and folders, and editing content



GitHub is an internet hosting service for software development and version control using Git

👉 When can each be useful?

# The importance of GitHub 🚀

- 100 million developers
- 372 million repositories (28 million are public)
- More than 20,000 academic papers are referenced in GitHub repositories

The screenshot shows the GitHub profile page for the 'Ersilia Open Source Initiative'. The header includes the GitHub logo, the username 'ersilia-os', a search bar, and navigation links for Overview, Repositories (229), Projects (2), Packages, Teams (3), People (16), and Settings. The profile section features a green repository icon, the name 'Ersilia Open Source Initiative', a bio stating it's a charity for global health drug discovery, and links to 110 followers, the United Kingdom location, the website 'http://ersilia.io', and social media handles '@ersiliaio' and 'hello@ersilia.io'. Below this is the 'README.md' file content, which includes a 'Welcome to Ersilia' message with a virus icon and a list of bullet points describing the organization's mission and resources. On the right, there's a 'View as: Public' dropdown, a note about viewing the README as a public user, a link to 'Get started with tasks', and a 'Discussions' section with a prompt to set up discussions.

ersilia-os

Search Type to search

Overview Repositories 229 Projects 2 Packages Teams 3 People 16 Settings

**Ersilia Open Source Initiative**

Ersilia is a charity developing open source tools to facilitate global health drug discovery, with a focus on neglected diseases, for equal healthcare

110 followers United Kingdom http://ersilia.io @ersiliaio hello@ersilia.io

Unfollow

README.md

**Welcome to Ersilia** 💊

- 🏠 The [Ersilia Open Source Initiative](#) is a **non-profit organization** dedicated to promoting open-source science and fostering a collaborative community.
- 🌍 Our mission is to **equip laboratories in the Global South** with machine learning and data science tools for drug discovery, empowering them to address local and global health challenges.
- 🧑‍🔬 Browse the [Ersilia Model Hub](#), our repository of **pre-trained machine learning models**!
- 📖 Visit more extensive **documentation** in the [Ersilia Book](#).

View as: **Public** ▼

You are viewing the README and pinned repositories as a public user.

[Get started with tasks](#) that most successful organizations complete.

Discussions

Set up discussions to engage with your community!

# Installing Git on a Windows computer

- Get the latest Git: <https://git-scm.com/download/win>
- Double-click to install the downloaded file
- Adjust your PATH environment by selecting “Run Git from the Windows Command Prompt”
- Configure line ending by selecting “Checkout Windows-style, commit UNIX-style line endings”

# Installing Git on a Linux computer

```
# Git is usually preinstalled in Linux
~/Desktop $ git
usage: git [--version] [--help] [-C <path>] [-c
<name>=<value>]
           [--exec-path[=<path>]] [--html-path]
           [--man-path] [--info-path]
           [-p | --paginate | -P | --no-pager]
           [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-
tree=<path>] [--namespace=<name>]
           [--super-prefix=<path>] [--config-
env=<name>=<envvar>]
           <command> [<args>]

These are common Git commands used in various
situations:
start a working area (see also: git help
tutorial)
clone Clone a repository into a new directory
init Create an empty Git repository or
reinitialise an existing one
work on the current change (see also: git help
everyday)
add Add file contents to the index
mv Move or rename a file, a directory, or a
symlink
restore Restore working tree files
...
```

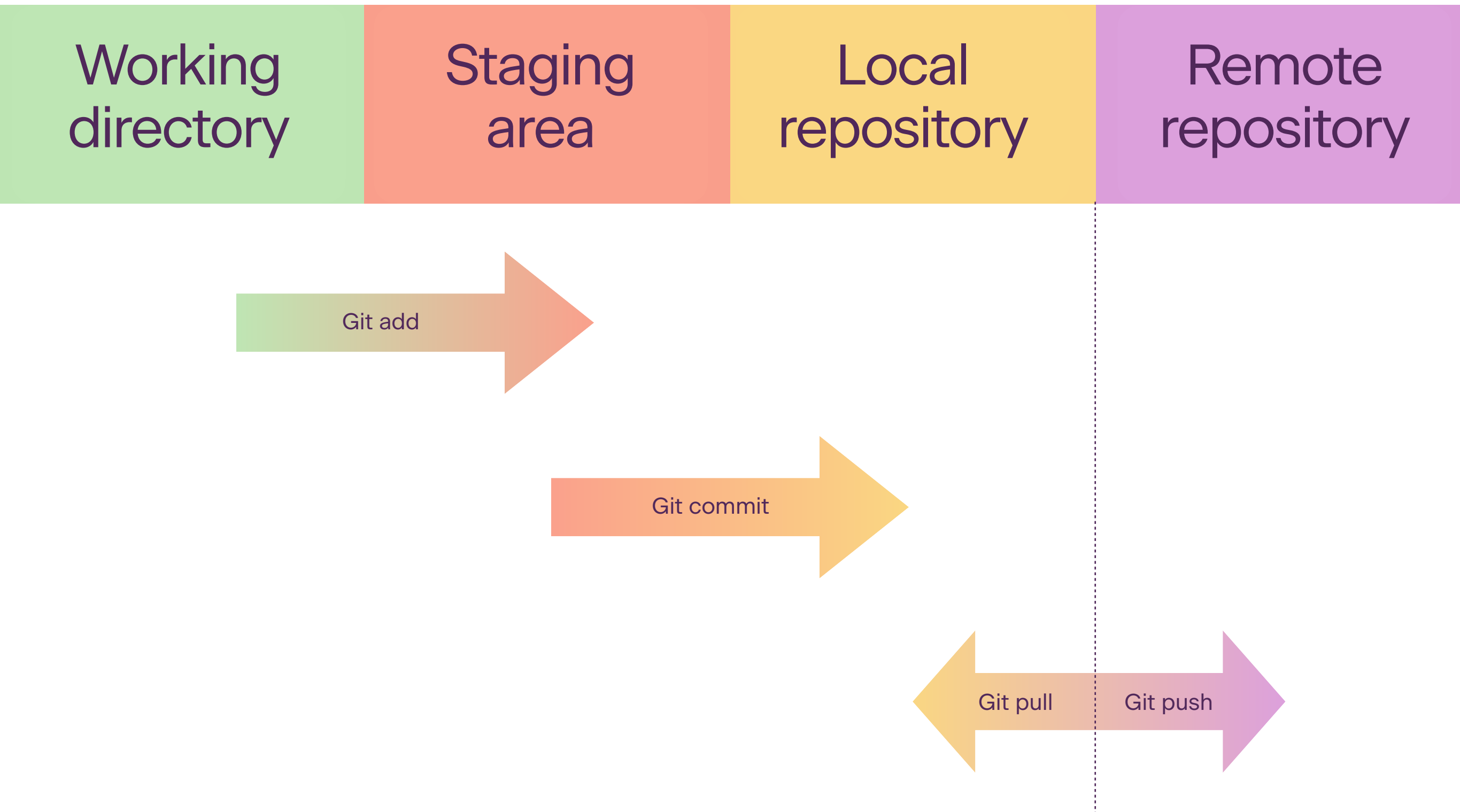
— Run the Git command in the terminal to make sure it is installed

## Installing Git on a MacOSX computer

- Check the install instructions: <https://git-scm.com/download/mac>
- Get Homebrew if you don't have it already
- In the Terminal, type: `brew install git`
- Check that Git works by typing the `git` command in the CLI



# How does Git work?



# Configure Git

## Use the git command in the CLI

```
#first, let's check we have Git properly installed
```

```
~/Desktop $ git --version
```

```
git version 2.39.2 (Apple Git-143)
```

```
~/Desktop $
```

```
#Let's set our username (same as GitHub user)
```

```
~/Desktop $ git config --global user.name "Gemma Turon"
```

```
~/Desktop $ git config --global user.email "gemma@ersilia.io"
```

# Start using Git

## Explore the git subcommands (init, status...)

```
# Let's navigate to our folder test
```

```
~/Desktop $ cd test
```

```
# Convert the folder into a Git – TRACKED folder
```

```
~/Desktop/test $ git init
```

```
Initialized empty Git repository in /Users/username/Desktop/test/.git
```

```
# Check what is Git following in the folder
```

```
~/Desktop/test $ git status
```

```
On branch main
```

```
No commits yet
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will be committed)
```

```
dates.txt
```

```
experiments/
```

```
results/
```

```
nothing added to commit but untracked files present (use "git add" to track)
```

# Start using Git

## Explore the git subcommands (init, status...)

```
# Let's add a file to track
```

```
~/Desktop/test $ git add dates.txt
```

```
~/Desktop/test $ git status
```

```
On branch main
```

```
No commits yet
```

```
Changes to be committed:
```

```
(use "git rm --cached <file> ..." to unstage)
```

```
new file: dates.txt
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will be committed)
```

```
experiments/
```

```
results/
```

```
nothing added to commit but untracked files present (use "git add" to track)
```

# Start using Git

## Explore the git subcommands (init, status...)

```
# Let's add everything as tracked
```

```
# we use the dot (.) to indicate all
```

```
~/Desktop/test $ git add .
```

```
~/Desktop/test $ git status
```

```
On branch main
```

```
No commits yet
```

```
Changes to be committed:
```

```
(use "git rm --cached <file> ..." to unstage)
```

```
new file: dates.txt
```

```
new file: experiments/dates.txt
```

```
new file: experiments/exp1.txt
```

```
new file: experiments/exp2.txt
```

```
new file: results/res1.txt
```

```
new file: results/res2.txt
```

# Start using Git

## Explore the git subcommands (init, status...)

```
# Once everything is in the staging area, we can commit the changes
```

```
# We add a message to the commit so we which changes we are making
```

```
~/Desktop/test $ git commit -m "my first commit"
[main (root-commit) 184ba96] my first commit
7 files changed, 0 insertions (+), 0 deletions (-)
create mode 100644 dates.txt
create mode 100644 experiments/dates.txt
create mode 100644 experiments/exp1.txt
create mode 100644 experiments/exp2.txt
create mode 100644 results/res1.txt
create mode 100644 results/res2.txt
~/Desktop/test $ git status
On branch main
nothing to commit, working tree clean
```



# Start using Git in the CLI

```
# Let's modify a file and see how we can track the changes
~/Desktop/test $ echo "my first line">dates.txt
~/Desktop/test $ git status
On branch main
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in workign directory)
modified:   dates.txt

no changes added to commit (use "git add" and/or "git commit -a")
~/Desktop/test $ git add .
~/Desktop/test $ git commit -m "trying changes"
[main fe3a311] trying changes
1 file changed, 1 insertion(+)

~/Desktop/test $ git status
On branch main
nothing to commit, working tree clean
```

# Start using Git

## Explore the git subcommands (init, status...)

```
# Let's see how the history of changes is recorded
~/Desktop/test $ git log
commit fe3a3115944350b1045047aeb2cecd4a23d68e (HEAD -> main)
Author: Gemma Turon (gemma@ersilia.io)
Dte: Wed Sep 6 19:10:00 2023 +0200

trying changes

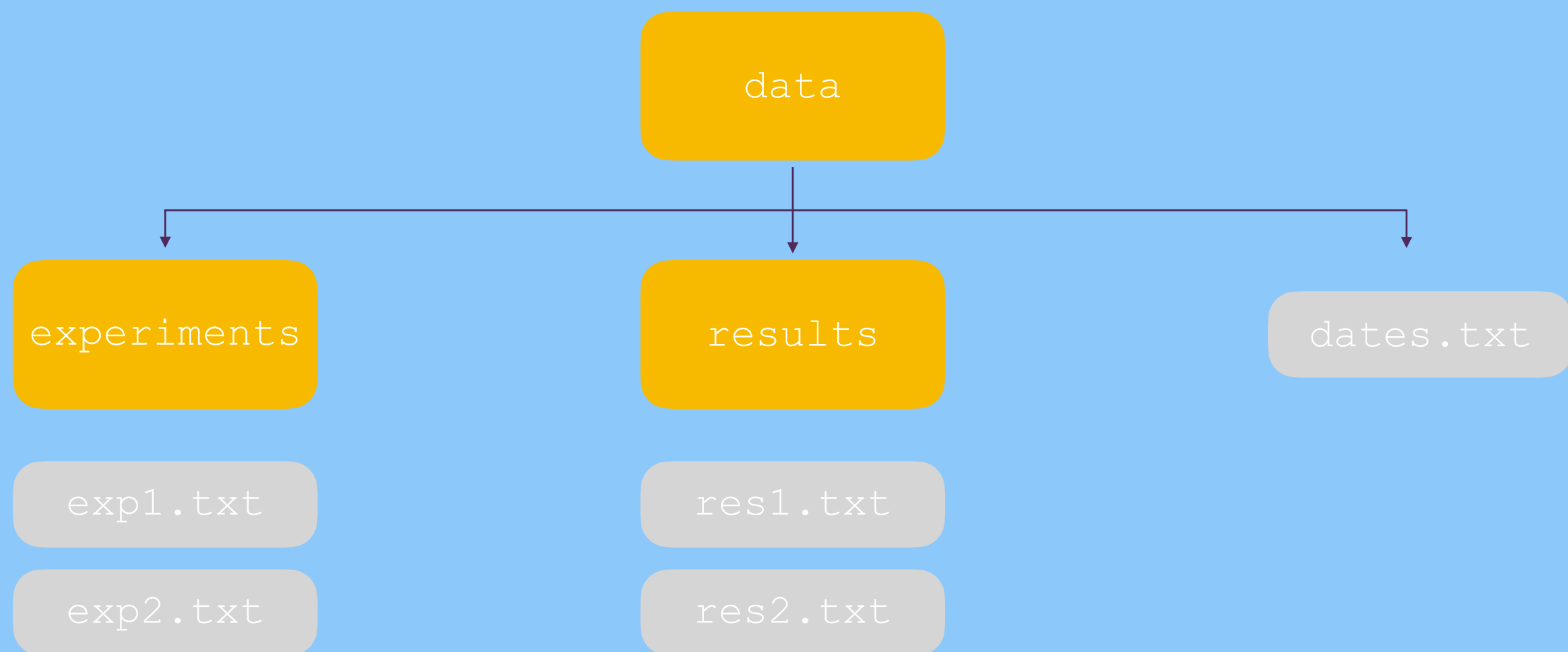
commit 184ba96df76f18097d2762c0d0596096455f621a
Author: Gemma Turon (gemma@ersilia.io)
Dte: Wed Sep 6 19:00:00 2023 +0200

first commit
```



# Practise Git commands!

- Include some changes in your files manually or through the CLI and add them to the Git repository



# Create a new GitHub repository

New

GemmaTuron

Overview


Repositories4

Projects

Packages

Stars5

Type to search



gemmaturon

GemmaTuron · she/her

@ersilia-os co-founder and CEO

Edit profile

51 followers · 4 following

@ersilia-os

gemma@ersilia.io

https://ersilia.io

@TuronGemma

in/gemma-turon

Popular repositories

ResourcesPublic

Forked from KairoiAI/Resources

Resources made available under CC-BY 4.0 by Kairoi

test-repoPublic

2,631 contributions in the last year

Contribution settings

SepOctNovDecJanFebMarAprMayJunJulAugSep

Mon

Wed

Fri

Learn how we count contributions

LessMore

Contribution activity

2023

2022

2021

2020

September 2023

Created 33 commits in 7 repositories

Created 1 repository

GemmaTuron/test-repo

# Create a new GitHub repository

Private vs Public?

.gitignore: git ignores certain types of files created by the computer (i.e python-related files)

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).


Required fields are marked with an asterisk (\*).

### Repository template

No template ▾

Start your repository with a template repository's contents.

### Owner \*

 GemmaTuron ▾

### Repository name \*

/

Name: no spaces, use hyphen to join words, for example test-repo

Great repository names are short and memorable. Need inspiration? How about [psychic-waffle](#) ?

### Description (optional)



### Public

Anyone on the internet can see this repository. You choose who can commit.



### Private

You choose who can see and commit to this repository.

### Initialize this repository with:



### Add a README file

This is where you can write a long description for your project. [Learn more about READMEs](#).

README: project descriptions

### Add .gitignore

.gitignore template: None ▾

Choose which files not to track from a list of templates. [Learn more about ignoring files](#).

### Choose a license

License: None ▾

License: who can use this code and how?

A license tells others what they can and can't do with your code. [Learn more about licenses](#).

 You are creating a public repository in your personal account.

Create repository

# Clone the repository in your local (directly as a Git folder)

GemmaTuron / test-repo

Q Type / to search

+ > ?

Issues Pull requests Actions Projects Wiki Security Insights Settings

test-repo

Public

Pin

Unwatch 1

Fork 0

Star 0

main 1 branch 0 tags

GemmaTuron Initial commit

.gitignore Initial commit

LICENSE Initial commit

README.md Initial commit

README.md

test-repo

Go to file

Add file

<> Code

Local

Codespaces

Clone

HTTPS SSH GitHub CLI

https://github.com/GemmaTuron/test-repo.g

Use Git or checkout with SVN using the web URL.

Open with GitHub Desktop

Download ZIP

About

No description, website, or topics provided.

Readme

Activity

0 stars

1 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package



# Clone the repository in your local (directly as a Git folder)

```
# Let's go to our Desktop and clone the repository there
```

```
$ cd Desktop
```

```
# Use Https
```

```
~/Desktop $ git clone https://github.com/GemmaTuron/test-repo.git
```

```
# Use SSH
```

```
~/Desktop $ git clone git@github.com:GemmaTuron/test-repo.git
```

```
Cloning into "test-repo" ...
```

```
remote: Enumerating objects: 5, done.
```

```
remote: Counting objects: 100% (5/5), done.
```

```
remote: Compressing objects: 100% (4/4), done.
```

```
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
```

```
Receiving objects: 100% (5/5), done.
```

```
~/Desktop $ ls
```

```
test-repo
```

```
test
```

```
~/Desktop $ cd test-repo
```

```
LICENSE      README.md
```

# Make local changes

```
~/Desktop/test-repo $ echo "this is a test" > README.md
~/Desktop/test-repo $ git status
On branch main
Your branch is up to date with "origin/main".

Changes not staged for commit:
(use "git add <file> ..." to update what will be committed)
(use "git restore <file> ..." to discard changes in working directory)
modified: README.md

no changes added to commit (use "git add" and/or "git commit -a")

# Let's add the modified file to the staging area
~/Desktop/test-repo $ git add README.md
~/Desktop/test-repo $ git commit -m "modifying readme"
[main e79b320] modifying readme
1 file changed, 1 insertion(+), 1 deletion(-)
~/Desktop/test-repo $ git status
On branch main
Your branch is ahead of "origin/main" by 1 commit.
(use "git push" to publish your local commits)
```

# Push changes to remote end & go online to see what happened!

```
# Let's push the changes to the remote end
~/Desktop/test-repo $ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 10 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 267 bytes | 267.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:GemmaTuron/test-repo.git
9f83f39..e79b320 main -> main
```


# Updated GitHub repository 🎉

GemmaTuron / test-repo

Q Type / to search

+ > 🔍

Issues Pull requests Actions Projects Wiki Security Insights Settings

 test-repo

Public

Pin


Unwatch 1

Fork 0




Star 0

main 1 branch 0 tags

Go to file Add file <> Code

 GemmaTuron modifying readme

e79b320 4 minutes ago ⌚ 2 commits

 .gitignore	Initial commit	24 minutes ago
 LICENSE	Initial commit	24 minutes ago
 README.md	modifying readme	4 minutes ago

README.md

this is a test

About

No description, website, or topics provided.

Readme

MIT license

Activity

0 stars

1 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

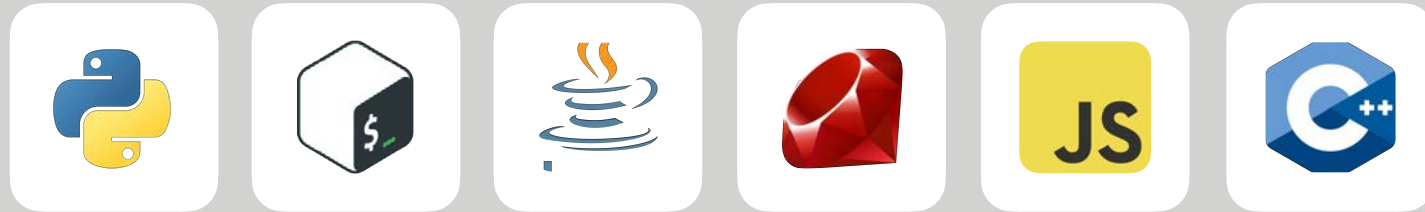


## Practise Git commands!

- Go to your online repository and add a file directly by using the “Add File” button
- Explore the `git pull` command to get a copy of this new file in your local repository as well!

# The Python programming language

# Multiple programming languages



Programming languages are sets of rules that convert human-readable text to machine code

Their instructions are precise and only have one single meaning



- Human readable syntax
- Flexibility for several applications
- Lots of pre-built packages



# Basic Python Concepts

- Variable: to store information, for example

```
x = 5
```

- Data Types:

- Numeric: `int` (integer) or `float` (decimal)

- Text: `"string"`

- Lists: ordered list of items (strings, integers...) for example

```
["one", "two", "three"]
```

```
[1, 2, 3]
```

- Functions: reusable blocks of code to perform a specific task:

```
def sum_numbers(x, y):  
    return (x + y)
```

# Python packages

```
# Install package via manager (pip or conda)

!pip install rdkit

# Import modules or functions from the package

import rdkit
from rdkit import Chem

# Call a specific function to use it
molecule =
Chem.MolFromSmiles("CCCCOCCCC")
```

Python packages are collections of reusable Python modules that provide pre-written code and functionality. They simplify code reuse and maintenance.

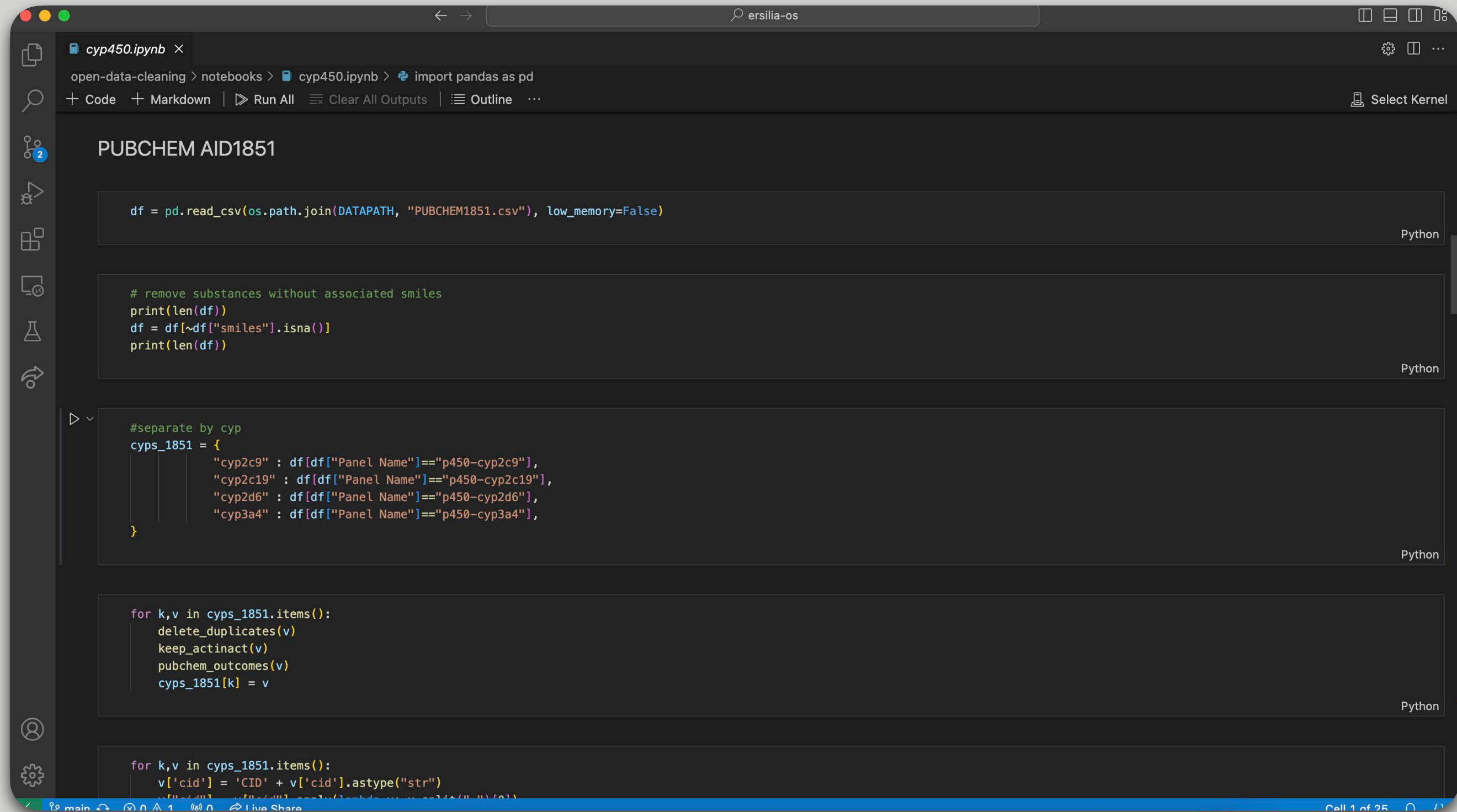
# Jupyter Notebooks

A notebook allows us to run small pieces of code at a time, making it very easy to perform data analysis.

Where can we run Jupyter Notebooks?

- Locally: we need python installed as well as a Jupyter system. The best is to do so via a Package Manager called Anaconda
- Remote computer: all the packages are installed in the remote computer and we access it via a CLI and/or GUI

# Data science with Jupyter Notebooks



The screenshot displays a Jupyter Notebook titled 'cyp450.ipynb' in a web browser. The notebook is open to a cell containing Python code for data cleaning and processing. The code is as follows:

```
df = pd.read_csv(os.path.join(DATAPATH, "PUBCHEM1851.csv"), low_memory=False)

# remove substances without associated smiles
print(len(df))
df = df[~df["smiles"].isna()]
print(len(df))

#separate by cyp
cyps_1851 = {
    "cyp2c9" : df[df["Panel Name"]=="p450-cyp2c9"],
    "cyp2c19" : df[df["Panel Name"]=="p450-cyp2c19"],
    "cyp2d6" : df[df["Panel Name"]=="p450-cyp2d6"],
    "cyp3a4" : df[df["Panel Name"]=="p450-cyp3a4"],
}

for k,v in cyps_1851.items():
    delete_duplicates(v)
    keep_actinact(v)
    pubchem_outcomes(v)
    cyps_1851[k] = v

for k,v in cyps_1851.items():
    v['cid'] = 'CID' + v['cid'].astype("str")
    v['cid'] = v['cid'].str.lstrip("0")
    v['cid'] = v['cid'].str.lstrip("0")
```

The notebook interface includes a sidebar with icons for file management, search, and other functions. The top bar shows the browser address 'ersilia-os' and a 'Select Kernel' button. The bottom status bar indicates 'Cell 1 of 25'.

# GitHub Codespaces

Codespaces is a cloud service provided by GitHub which gives users 60h of free cloud computing time per month.

We will use it throughout the course, why?

- Integrated within GitHub, where all our code resides
- Allows for preinstallation of packages

👉 Sign into your GitHub account and go to <https://github.com/ersilia-os/ersilia-intro-workshop>

# Let's recap!

In this module we have heard about...

- The command line interface (CLI)
- Git and GitHub
- Python and Jupyter Notebooks

 Go to the course repository!

 Try GitHub Codespaces!