Introduction to Jupyter notebooks

Case Study:

RESTful web service for solving multidimensional time-independent Schrödinger equation

"Ss. Cyril and Methodius" University - Skopje
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

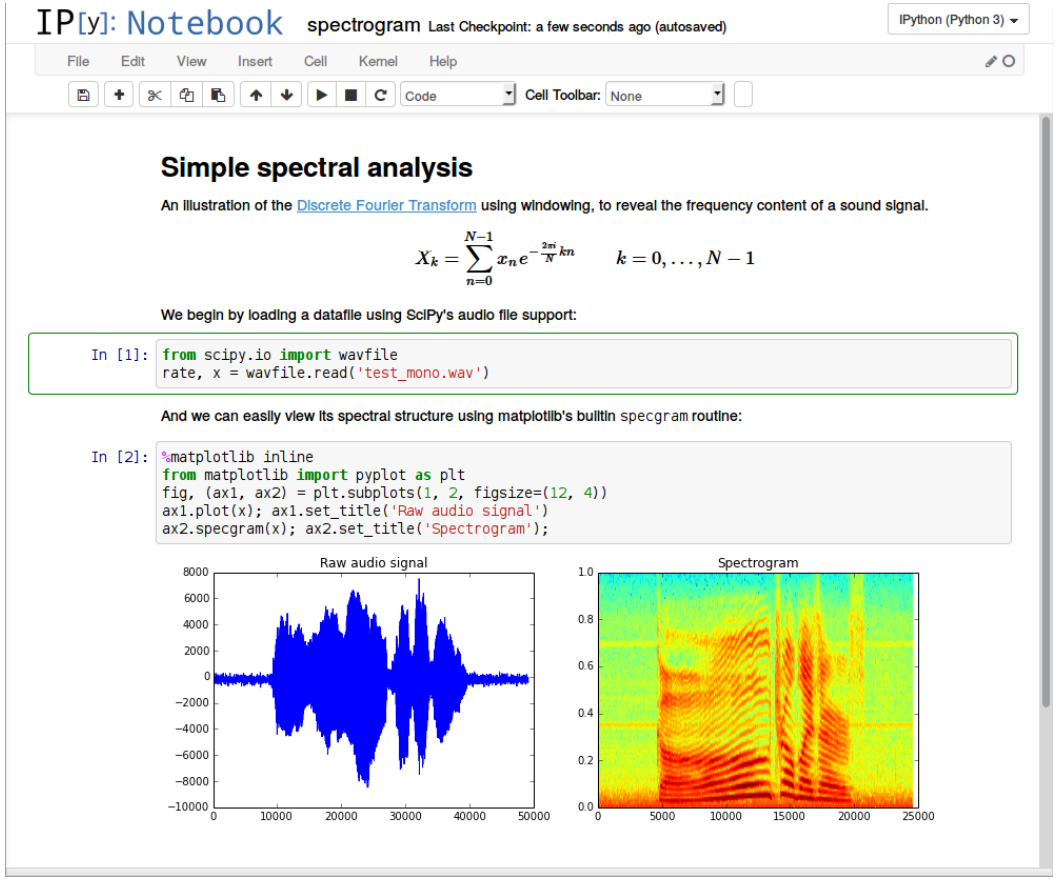
National Initiatives for Open Science in Europe – H2020 Research and Innovation action – contract no. 857645

What is a Notebook?

A *notebook* combines the functionality of

- a word processor — handles formatted text
- integrates code and its output into a single document
- combines visualizations, narrative text, mathematical equations, and other rich media

Single document where you can run code, display the output, add explanations, formulas, charts, and make your work more transparent, understandable, repeatable, and shareable.

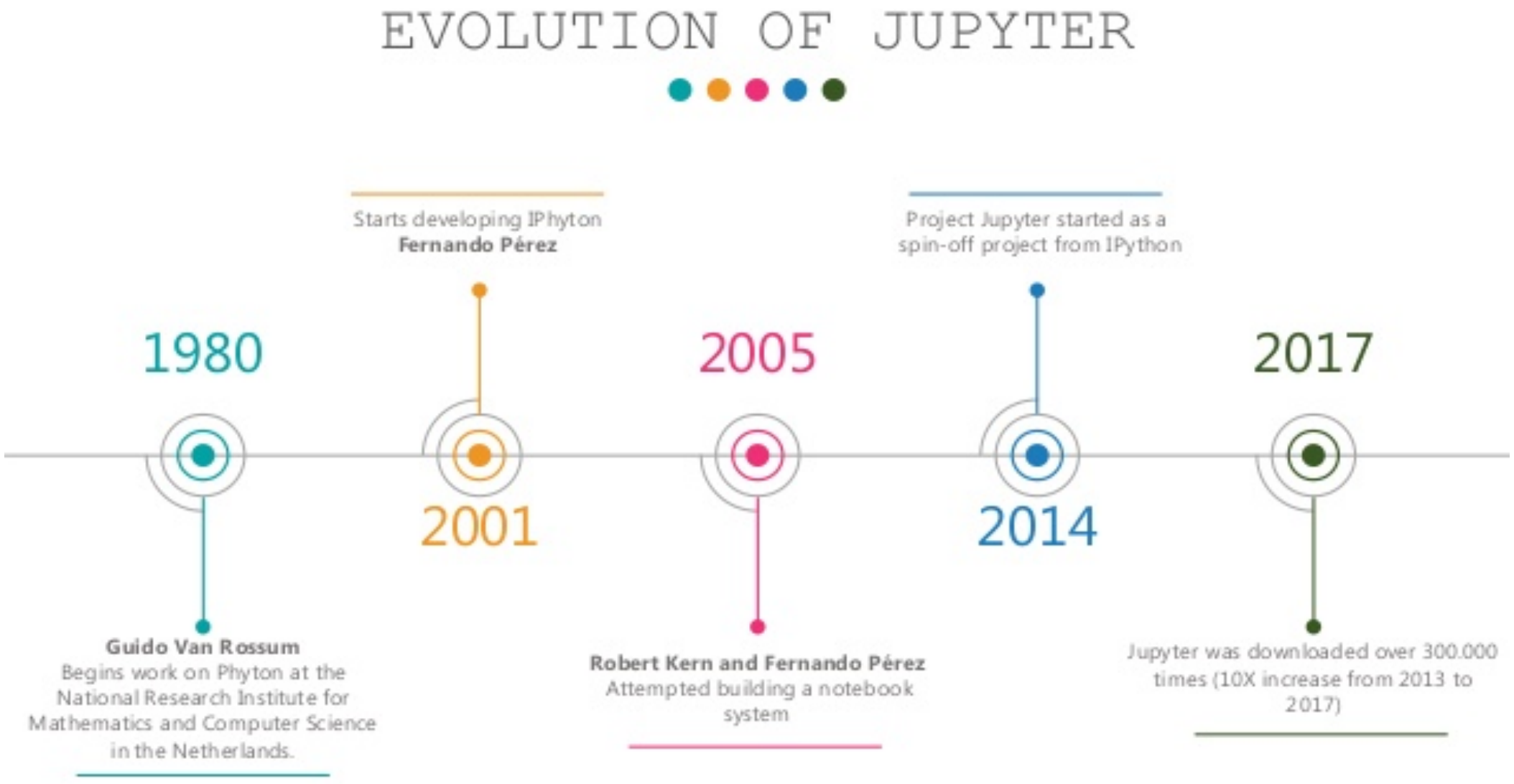


What is Jupyter Notebook?

- Incredibly powerful tool for interactively developing and presenting data science projects
- Open-source web application that you can use to create and share documents
- Can be installed locally (Python installed)
  - pip3 install jupyter (Linux, Mac OS x)
  - Anaconda Navigator (Windows)
- or run Online without installing anything

  https://jupyter.org/try

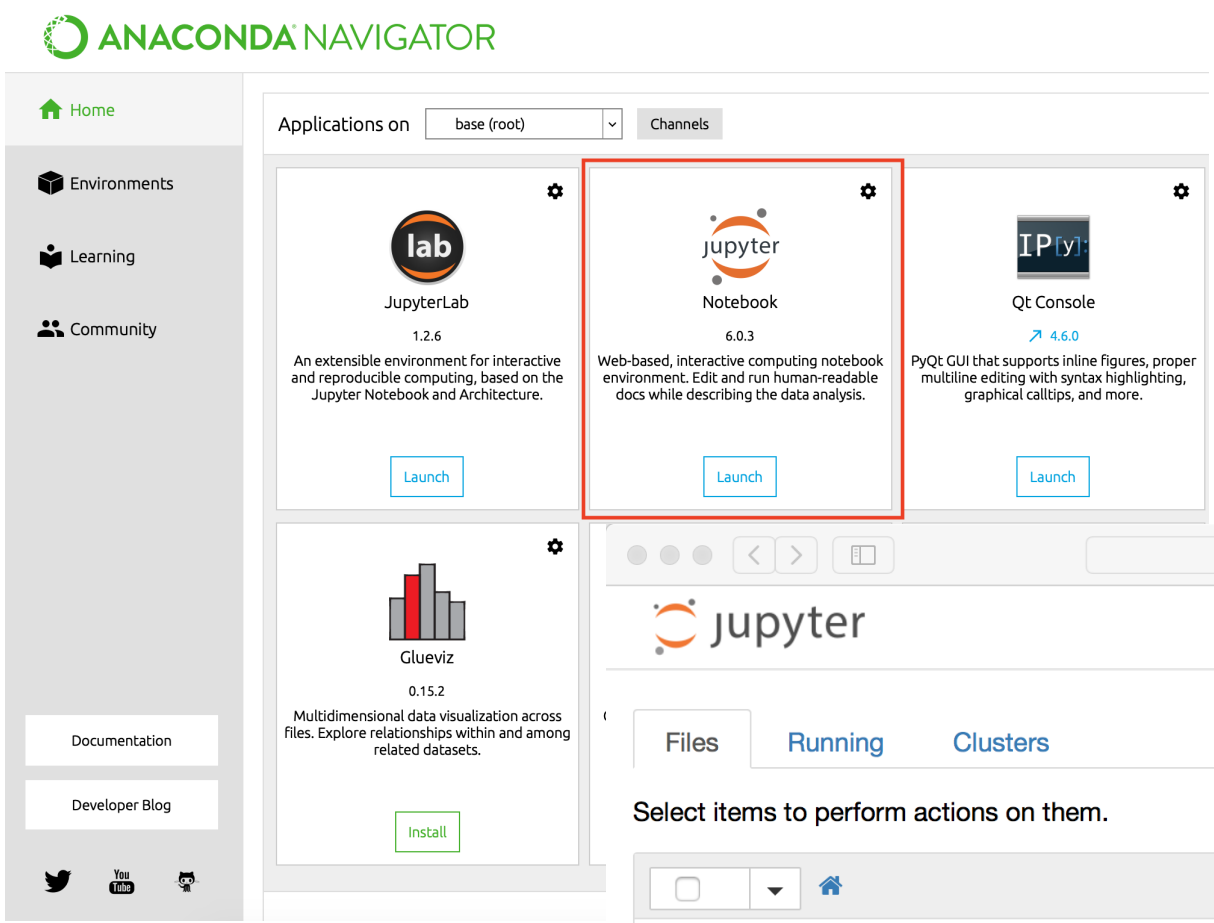
Evolution of Jupyter



Jupyter acronym

Jupyter is a loose acronym meaning **Julia**, **Python**, and **R**

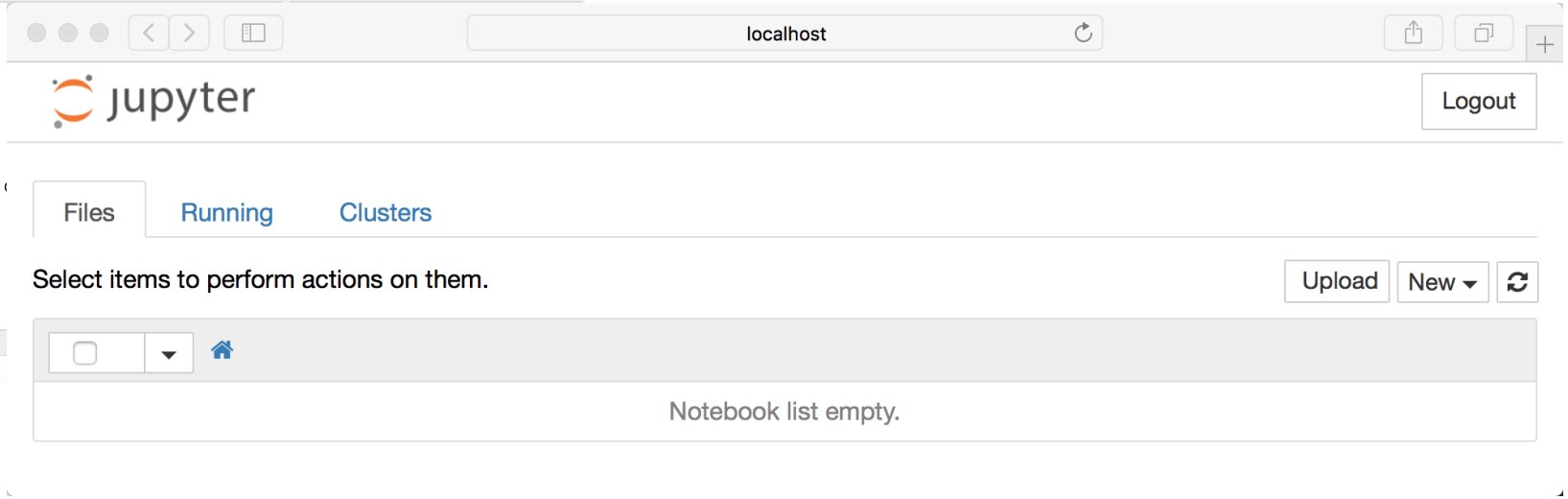
How to launch Jupyter Notebook



Shell

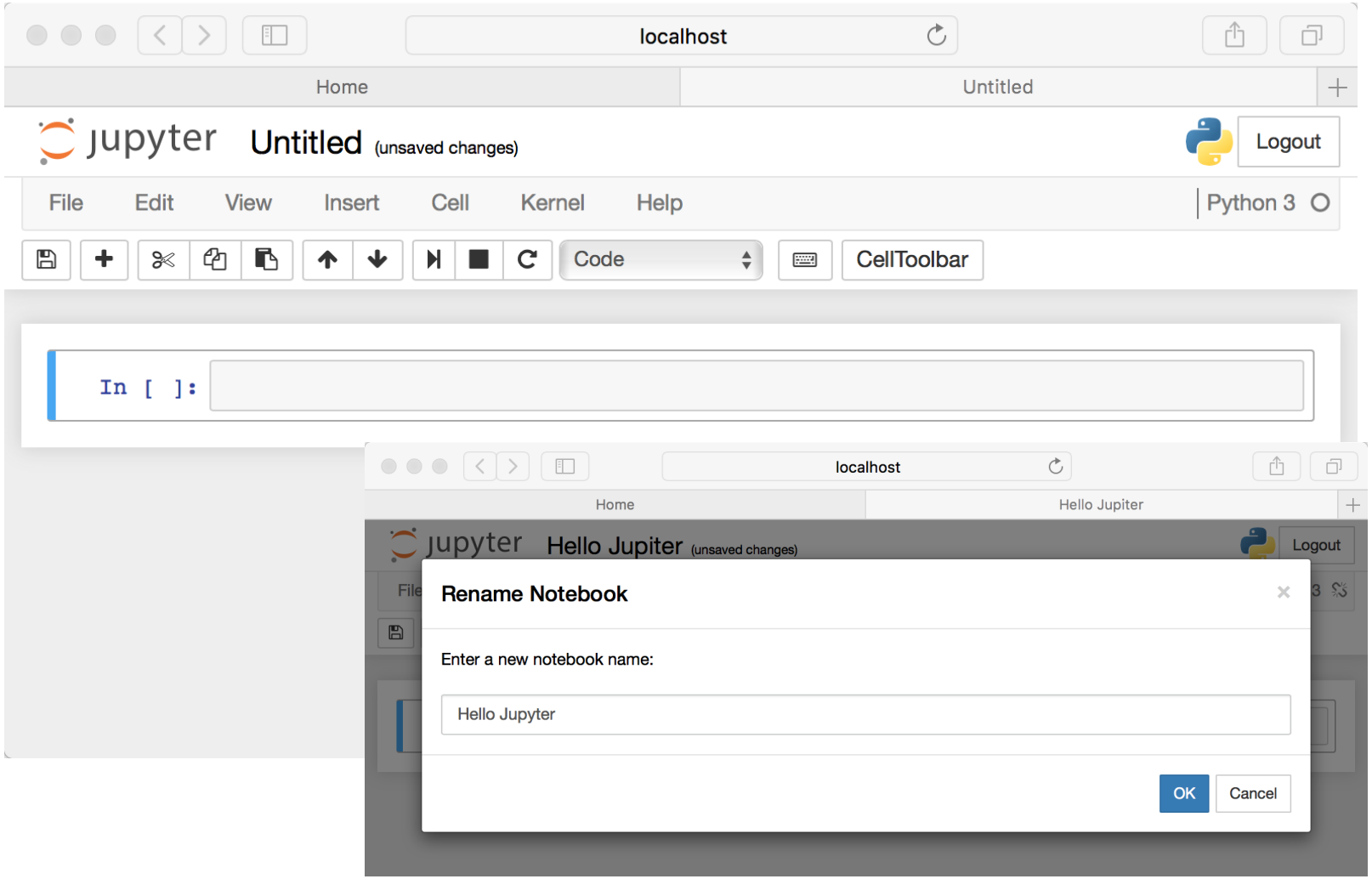
```
$ jupyter notebook
```

This will start up Jupyter and your default browser should start (or open a new tab) to the following URL: http://localhost:8888/tree

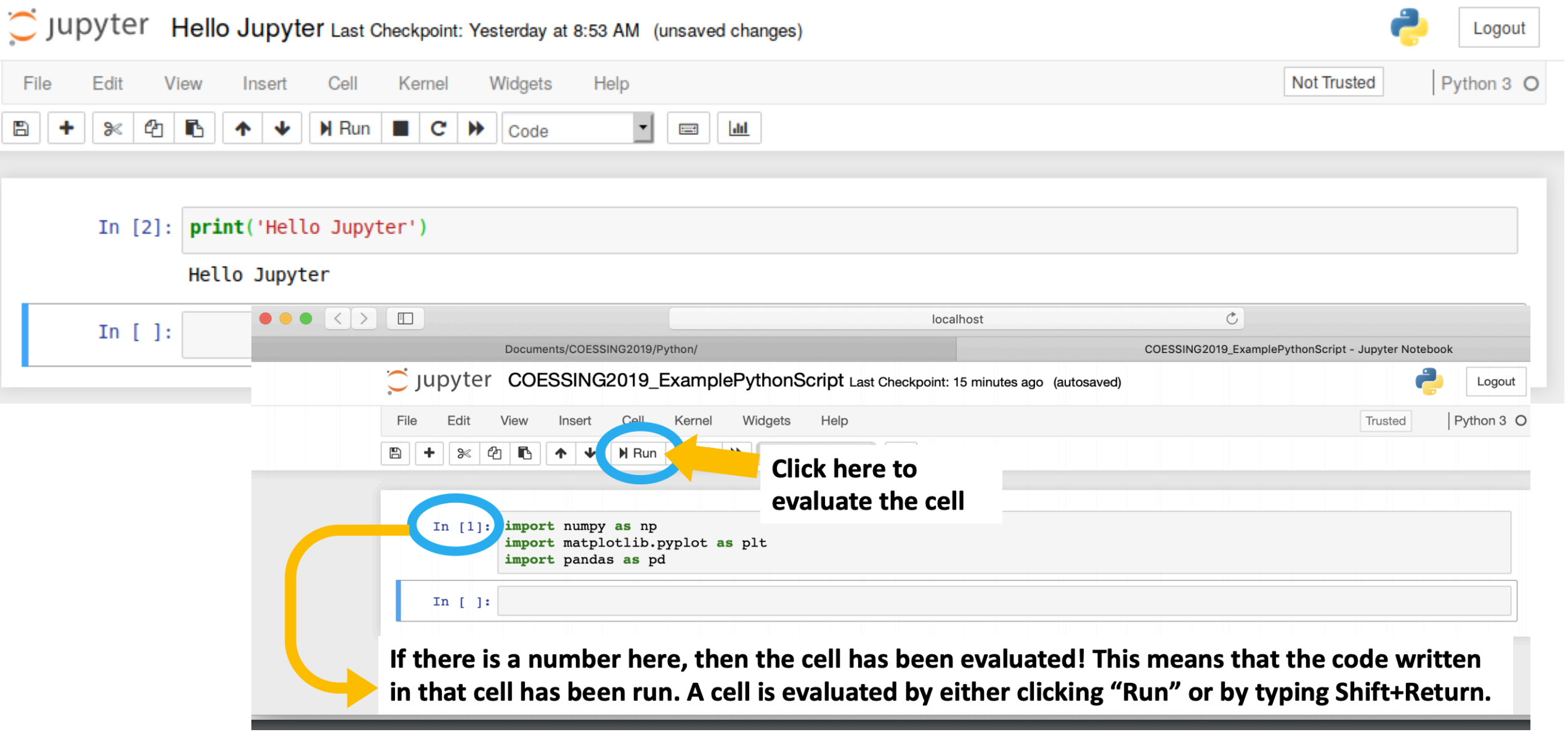


Creating Your First Notebook

- File-> New Notebook
- If you have Python 2 and Python 3 installed, you can create a Notebook that uses either of these
- If you switch back to the dashboard, you will see the new file Untitled.ipynb

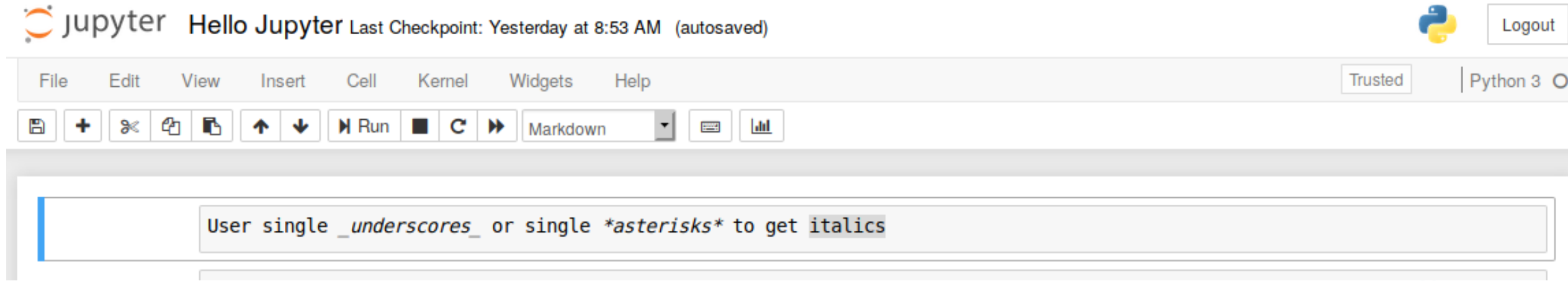


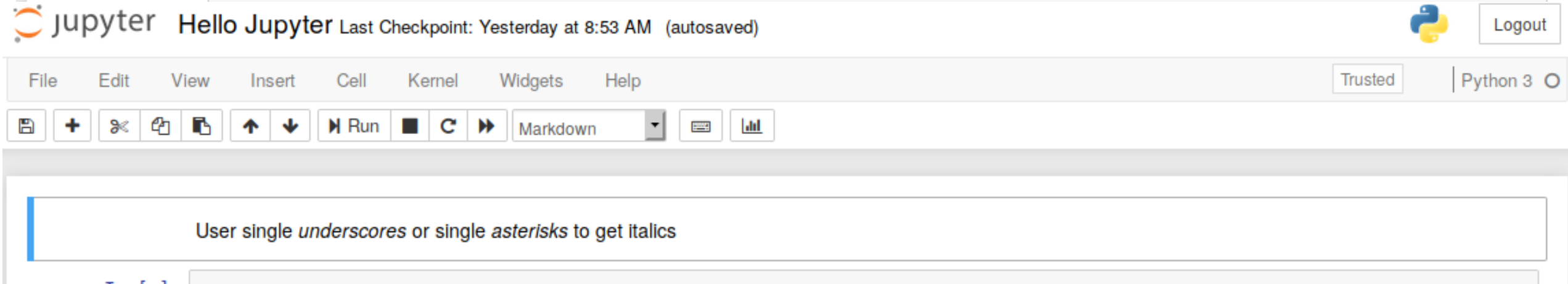
Running Cells



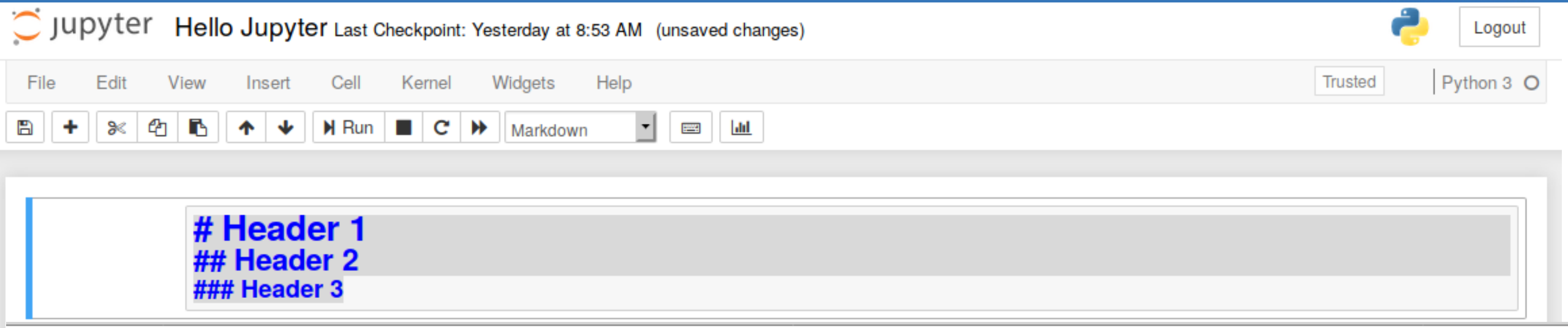
Cell Type

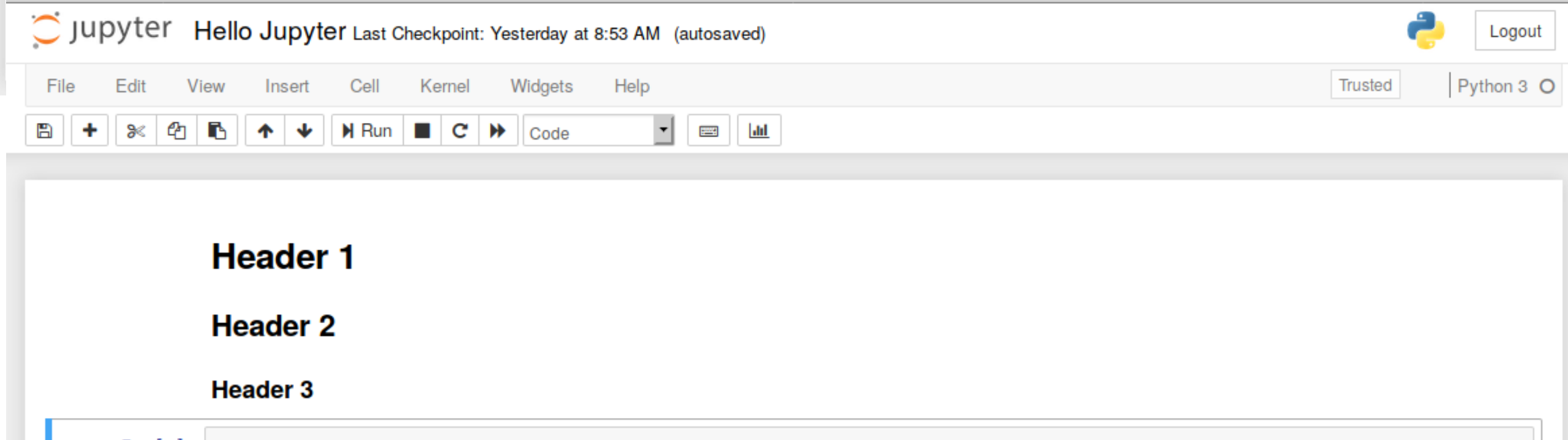
- Cell -> Cell Type
  - Code, Markdown and Raw NBConvert
- Markdown - markup language that is a superset of HTML.



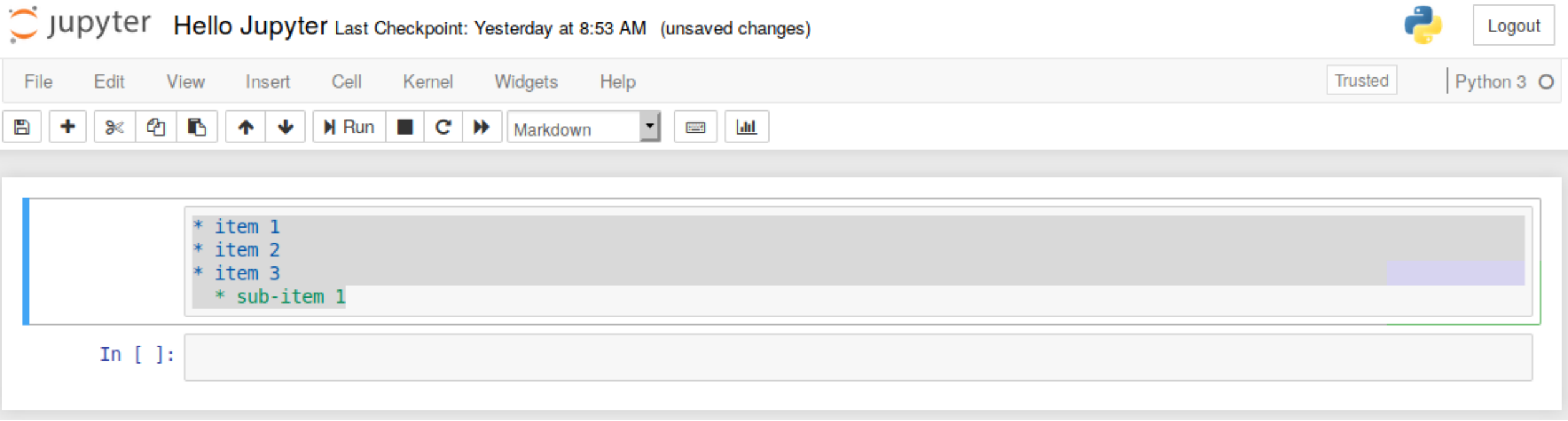


Headers


jupyter Hello Jupyter Last Checkpoint: Yesterday at 8:53 AM (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Markdown
Header 1
Header 2
Header 3

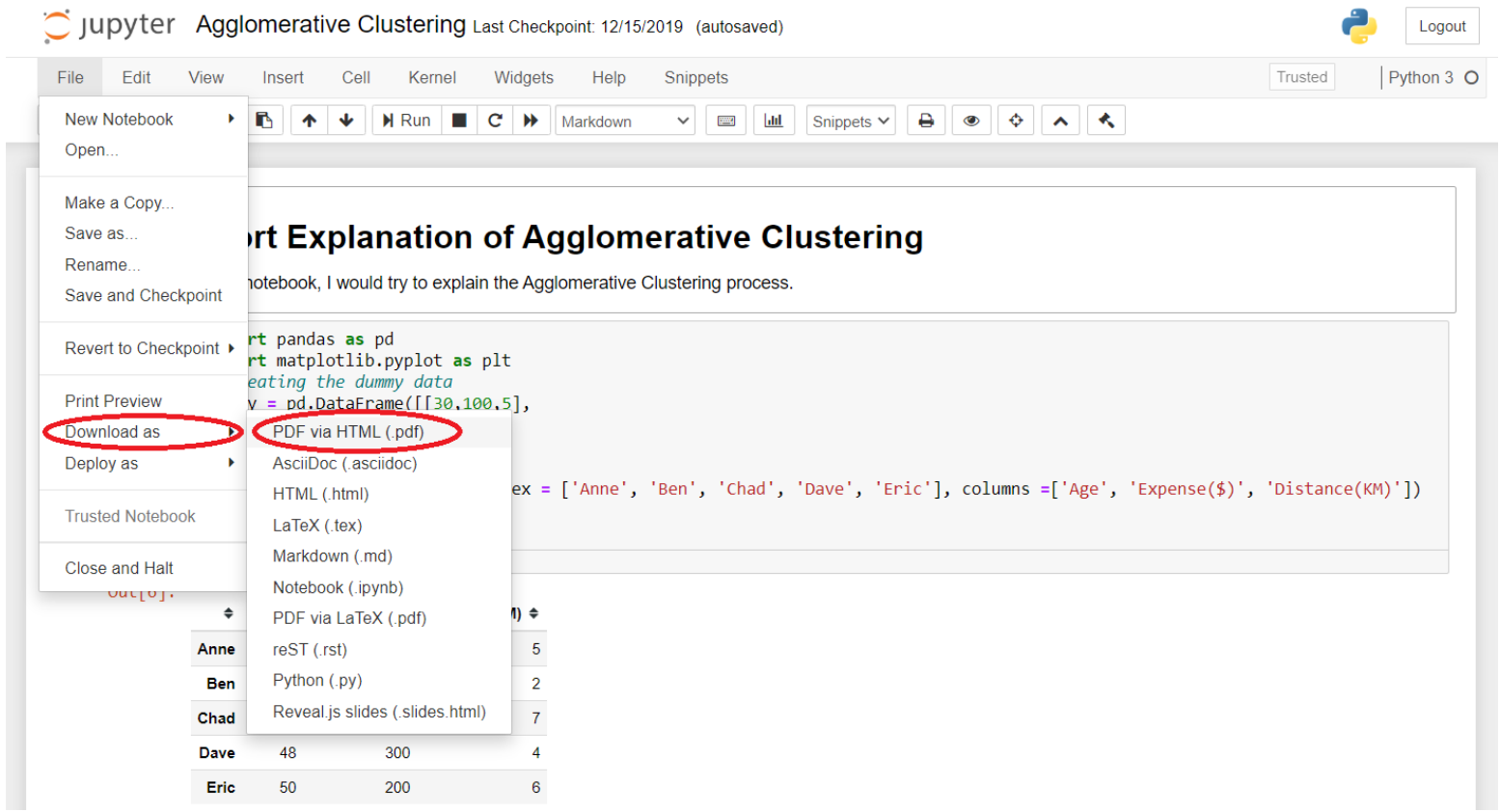

jupyter Hello Jupyter Last Checkpoint: Yesterday at 8:53 AM (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Header 1
Header 2
Header 3

Lists


jupyter
Hello Jupyter
Last Checkpoint: Yesterday at 8:53 AM
(unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Markdown
* item 1
* item 2
* item 3
* sub-item 1
In [ ]:

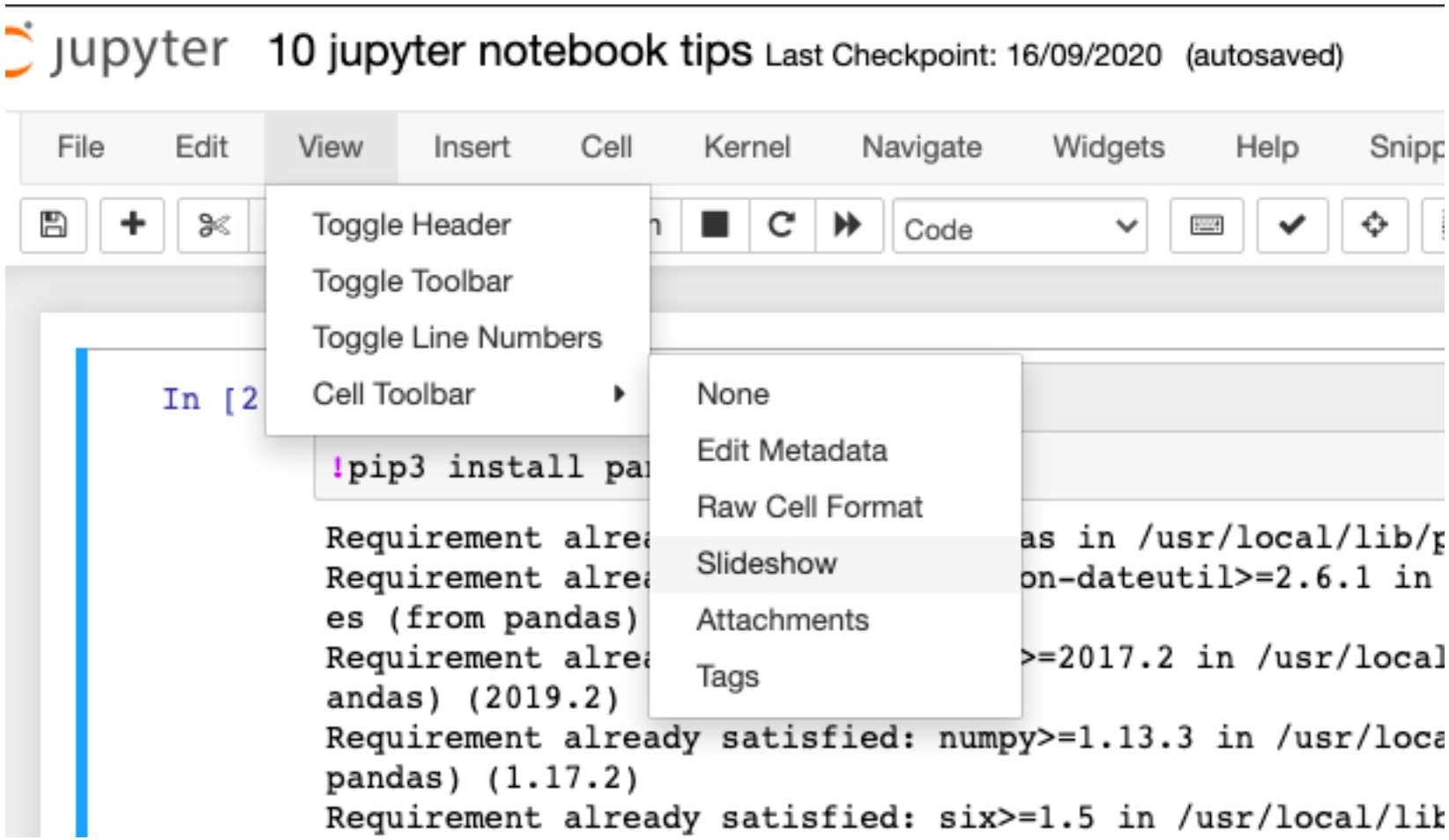
Exporting Notebooks

- HTML
- LaTeX
- PDF
- RevealJS
- Markdown
- ReStructured Text
- Executable script



Creating Presentations

View -> Cell Toolbar -> Slideshow



Creating Presentations

Once you have done this each cell should have a Slide Type option in the right-hand corner of the cell

In [4]:

Slide Typ

- ✓ Slide
- Sub-Slide
- Fragment
- Skip
- Notes

```
%matplotlib inline

import matplotlib
import matplotlib.pyplot as plt
import numpy as np

# Data for plotting sine function
t = np.arange(0.0, 2.0, 0.01)
s = 1 + np.sin(2 * np.pi * t)

plt.plot(t, s);
```

Examples

IP[y]: Notebook spectrogram Last Checkpoint: a few seconds ago (autosaved)

IPython (Python 3)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None

Simple spectral analysis

An illustration of the Discrete Fourier Transform using windowing, to reveal the frequency content of a sound signal.

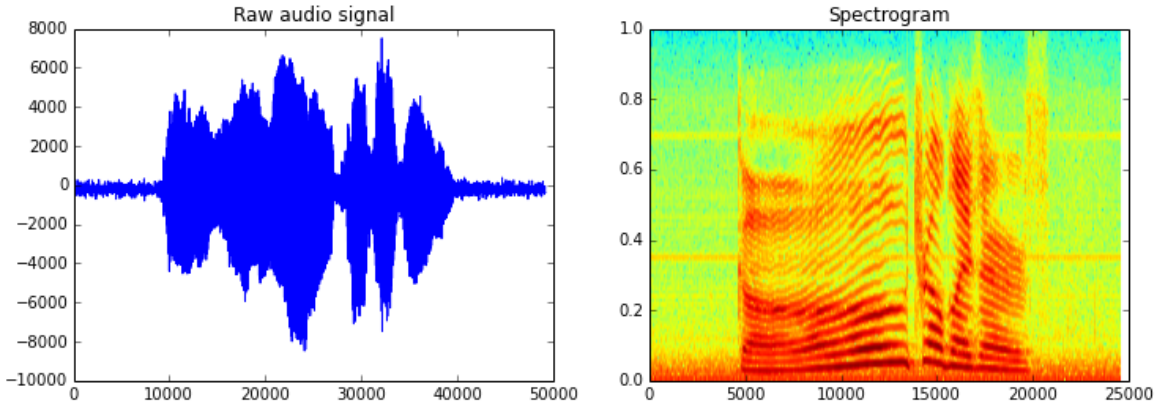
$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn} \qquad k = 0, \ldots, N-1$$

We begin by loading a datafile using SciPy's audio file support:

In [1]:
```
from scipy.io import wavfile
rate, x = wavfile.read('test_mono.wav')
```

And we can easily view its spectral structure using matplotlib's builtin `specgram` routine:

In [2]:
```
%matplotlib inline
from matplotlib import pyplot as plt
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 4))
ax1.plot(x); ax1.set_title('Raw audio signal')
ax2.specgram(x); ax2.set_title('Spectrogram');
```

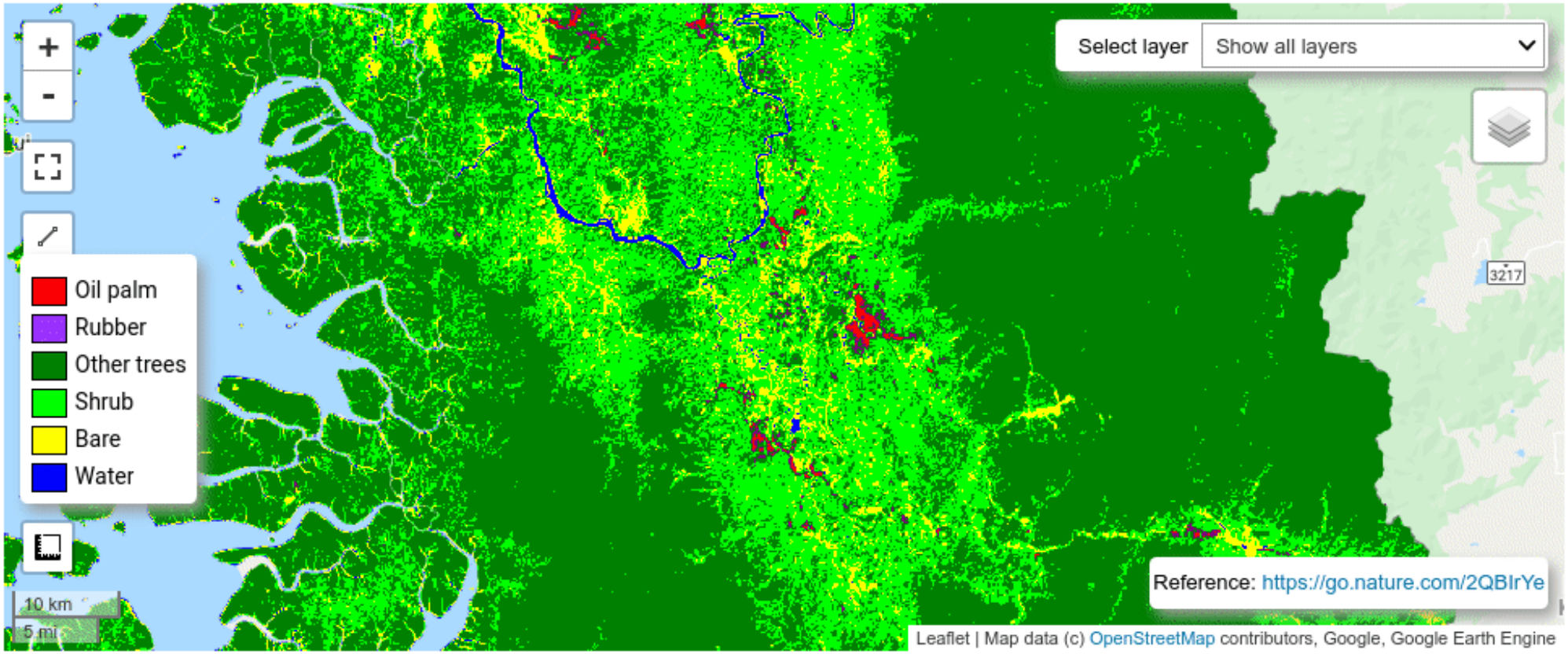


National Initiatives for Open Science in Europe – H2020 Research and Innovation action – contract no. 857645

Create an interactive map

The default basemap is `Google Satellite`. Additional basemaps can be added using the `Map.add_basemap()` function.

In [2]:
```
Map = geemap.Map(center=[40,-100], zoom=4)
Map.add_basemap('ROADMAP') # Add Google Map
Map
```



Home Page - Select or create a n | 02_Data_processing - Jupyter No

https://hub.gke.mybinder.org/user/slicer-slicernotebooks-ybnh0x9d/notebooks/02_Data_processing.ipynb#

jupyter 02_Data_processing (unsaved changes)

Visit repo | Copy Binder link

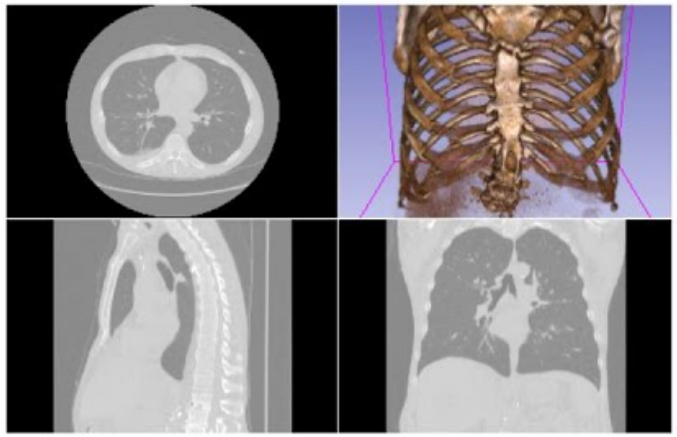
File Edit View Insert Cell Kernel Widgets Help | Not Trusted | Slicer 4.11

Run | Code

```
# Volume render a small (30%) image
slicernb.AppWindow.setWindowSize(scale=0.3)
slicernb.showVolumeRendering(volume)
slicernb.showSliceViewAnnotations(False)

display(slicernb.ViewDisplay("FourUp"))

# Disable volume rendering to make sure it does not slow down execution
slicernb.showVolumeRendering(volume, False)
```



In [ ]:

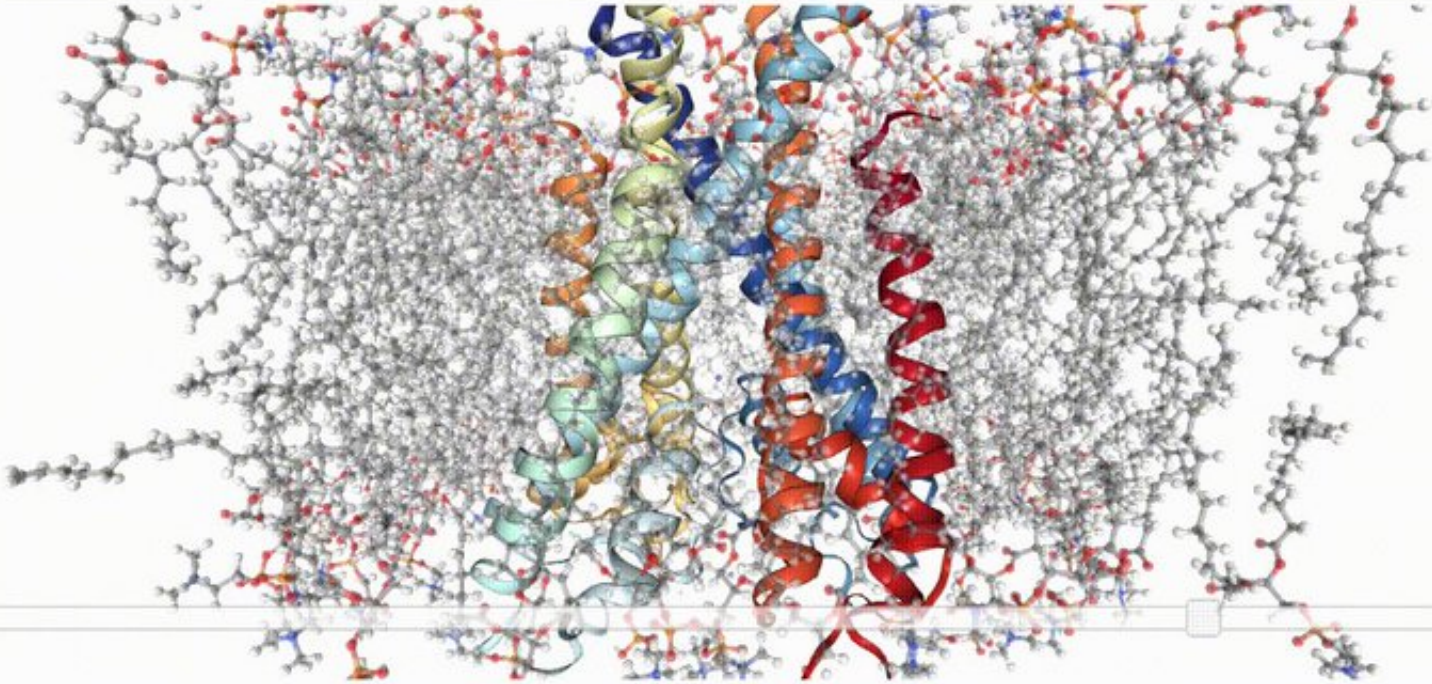
```
# Print all parameters of a CLI module
cliModule = slicer.modules.grayscalemodelmaker
n=cliModule.cliModuleLogic().CreateNode()
for groupIndex in range(n.GetNumberOfParameterGroups()):
  for parameterIndex in range(n.GetNumberOfParametersInGroup(groupIndex)):
    print('Parameter ({0}/{1}): {2} ({3})'.format(groupIndex, parameterIndex, n.GetParameterName(groupIndex, parameterIndex),
```

01_Data_loading_a....ipynb
Open file ...

Show all

In [1]:
```
import pytraj as pt
import nglview as nv
```

In [2]:
```
traj =  pt.load('sim.nc', top='sim.prmtop')
view = nv.show_pytraj(traj)
view
```



In [3]:
```
view.clear()
view.add_cartoon('protein', color_scheme='residueindex')
view.add_ball_and_stick('not protein', opacity=0.5)
```

Jupyter DEMO

A gallery of interesting Jupyter Notebooks:

https://github.com/jupyter/jupyter/wiki/A-gallery-of-interesting-Jupyter-Notebooks#programming-and-computer-science

Case Study

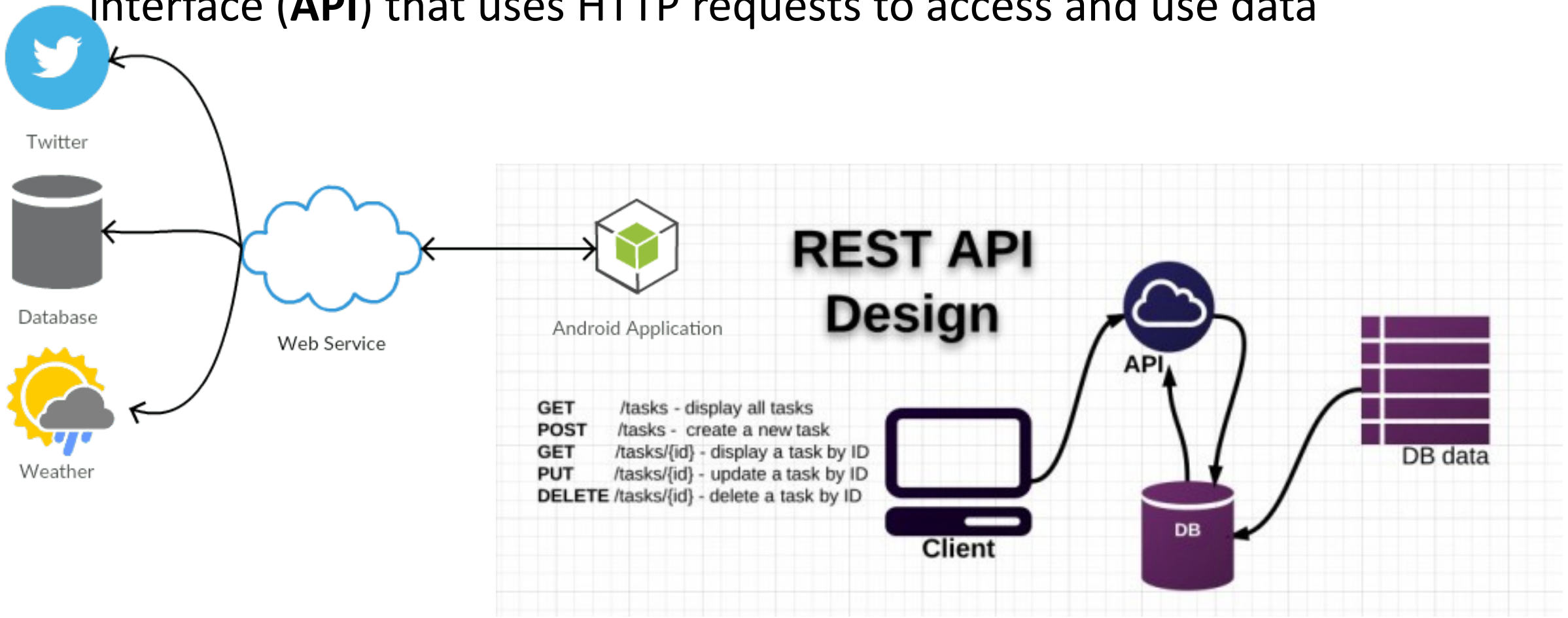
RESTful web service for solving multidimensional time-independent Schrödinger equation using Hermite DVR approach

- solution of **one-dimensional**, **two-dimensional** and **three-dimensional** time-independent Schrödinger equation based on the the Gauss-Hermite Discrete Variable Representation (DVR) approach

- The solution of one-dimensional Schrödinger equation is illustrated in the case of following model potentials:
  - Morse potential;
  - Simple Harmonic Oscillator (SHO) potential;
  - Sombrero potential (Mexican hat);
  - Woods-Saxon potential.
- Solutions of two-dimensional and three-dimensional Schrödinger equations are illustrated for the following two model potentials: multidimensional Morse potential and multidimensional SHO potential.

What is RESTful API?

- A **RESTful API** is an architectural style for an application program interface (**API**) that uses HTTP requests to access and use data



RESTful API Benefits

- **Separation between the client and the server**
- **Visibility, reliability and scalability**
- **The REST API is always independent of the type of platform or languages**
  - With a REST API you can have PHP, Java, Python or Node.js servers….
  - The only thing is that it is indispensable that the responses to the requests should always take place in the language used for the information exchange, normally XML or JSON.

Example: 1D Morse potential

- Returns a one-dimensional Morse potential V(x):
  - V(x) = D * (1 - exp(-a * (x - x0)))^2 – D
- Parameters:
  - **npts** - number of points (default value 10)
  - **D** - dissociation depth(default value 3.0)
  - **a** - inverse "width" of the potential (default value 0.5)
  - **x0** - equilibrium bond distance (default value 0.0)
  - **prec** - precision (default value 6)

How to consume RESTful API?

- You **type** http://194.149.135.58:8080/SchrodingerAPI/1dHermiteMorse into the address bar of your **browser**.

[-2.41671645 -1.39124794 -0.28535681 1.09633735 7.42311473]

- What if we want to use the results in our code?
  - Copy them? NO
  - Consume the RESTful web service in our code
    - DEMO in Jupyter

How to consume RESTful API?

Example with default parameters:

```
import requests
response = requests.get('http://194.149.135.58:8080/SchrodingerAPI/1dHermiteMorse')
if response.status_code == 200:
        print(response.content.decode('utf-8'))
else:
        print("None")
```

```
[-2.41671645 -1.39124794 -0.28535681  1.09633735  7.42311473]
```

Example with parameters (npts=10, D=3.0, a=0.5, x0=0.0, prec=6):

```
import requests
response = requests.get('http://194.149.135.58:8080/SchrodingerAPI/1dHermiteMorse?npts=10&D=3.0&a=0.5&x0=0.0&prec=6')
if response.status_code == 200:
        print(response.content.decode('utf-8'))
else:
        print("None")
```

```
[-2.418857 -1.441745 -0.643831  0.244456  1.40058   2.844225  4.900026  6.987975 19.116035 61.33091 ]
```

Schrödinger equation RESTful API DEMO

Why data scientists love Jupyter notebooks?

- **All in one place**
- **Easy to share**
- **Easy to convert**
- **Language independent**
- **Easy to customize**
- **Stress-free Reproducible experiments**
- **Effective teaching-cum-learning tool**
- **Interactive code and data exploration**

- One analysis of the code-sharing site GitHub counted more than 2.5 million public Jupyter notebooks in September 2018
- Jupyter was awarded the 2017 ACM Software Systems Award — a prestigious honor it shares with Java, Unix, and the Web.

How Netflix uses Jupyter Notebook?

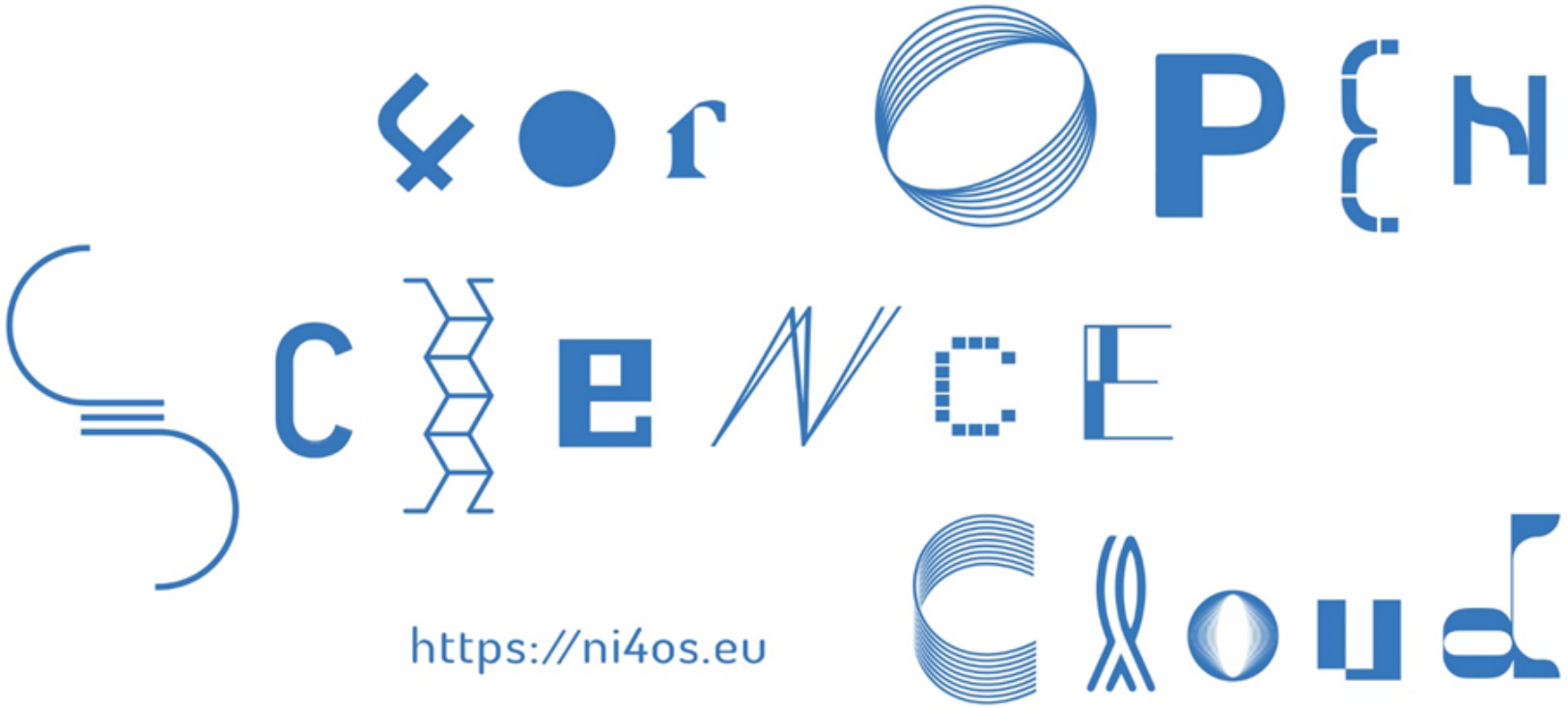
195 million

paid

- Notebooks were first introduced at Netflix to support data science workflows.
- To help the platform scale, they wanted to minimize the number of tools and the solution to this was the open-source tool: Jupyter notebooks.
- Users have programmatic access to virtually the entire platform from within a notebook.
- Netflix is going all-in on notebooks in production by migrating over 10k workflows to notebooks and using them as a way to bridge the chasm between technical and non-technical users

Contact Points

- Bojana Koteska
  - *bojana.koteska@finki.ukim.mk*
- Ljupco Pejov
  - *ljupcop@pmf.ukim.mk*
- Anastas Mishev
  - *anastas.mishev@finki.ukim.mk*



NI4OS eu

NI4OS

ni4os.eu