# 🪴🧑‍🏫 PyData NYC Nebari Tutorial - Speaker Notes

## Before we start

- Increase the font size so that the text visible to everyone
- Use light mode for presentation
- Cadence
    - introduce a topic
    - explain what they will do (and how it's useful)
    - demo
    - hands-on
- Make sure to have a water bottle on your desk

## Overview

- *welcome.slides.html*
- Introductions
    - Eskild – software engineer, nebari maintainer
    - Peyton - software engineer
- Key takeaways
    1. What is Nebari?
        - Previously known as "QHub"
        - An opinionated JupyterHub distribution (running on Kubernetes), managed by Terraform
        - Motivation to use Nebari
            - we've already spent the time (~2 years) to make these work together
        - **Walk through registration**
            - *If it's slow to start, explain why*
            - *We will keep the instance on for the weekend*
            - *Default group*
    2. Key features
    3. Deploy a Nebari cluster
        - **Start a cloud deployment**
            - *Don't worry about the details (we'll cover them later)*
            - *Deploying another Nebari cluster from within this existing Nebari cluster is NOT advised*
            - *Navigate to pydata.nebari.dev*
- *Show of hands if you haven't logged in / cloned repo*

# Interfaces

- Motivation
    - *Different people prefer to work in different ways*
    - *Shared folder makes it easier to collaborate*
        - *Keycloak controls access to these groups*
- Key takeaways
    - *Show of hands – folks who use JupyterLab*
    1. **Create a notebook (with a unique name) in the shared/analyst folder**
        - *Highlight the **shared folder** and how access to these folders is controlled*
            - *Highlight **examples** folder*
    - *Show of hands – folks who use VS Code*
    2. **Play with VS-code,**
        - *Add an extension*
            - *Note about this being different from Microsoft-served VS Code*
        - *Change env*
        - *Use debugger*
        - *Take a look at the hidden files*
    - *Show of hands - folks who have used SSH to remote into another machine*
    3. **SSH**
        - *show how to generate a JUPYTERHUB_API_TOKEN*
        - *If you want to work from your local vscode*

# Conda-store

- Peyton quick intro and background
- Most people are used to installing software on their laptops. In the cloud, in data platforms, usually you have less flexibility installing software that you want. Usually IT determines what software you can install, and there's often a request process for changes to the software you install. On top of that, a lot of JupyterHub distributions bake in the environment into the docker image, so changing the environment means a huge infrastructure effort to vet the package, rebuild the docker image, and redeploy to the cloud. **conda-store** is a tool that makes installing software easier. It gives users control over the software while being able to meet constraints imposed by IT.
- Try showing them the old UI
- Show the video at half speed.
- First feature: namespace management. If you're working on a project or as part of a group, there's ways to share environments.
- Trying to enforce best practices while still making it easy for the user to use.

- xarray example - these environments were created on the fly. Most JupyterHub deployments require you to specify the environment before deployment.

# Dask

- Dask is a library which allows you to parallelize python computing tasks easily – from your laptop or giant cluster on the cloud. If you're going to use a cloud computing service to run computations with Dask, you'll need Dask Gateway, which is a server that does cluster management for you when you submit a job.
- Let's jump into an example: we're going to load a dataset and run some computations on it, and in the process you'll get to see some of the cool dashboarding tools that Dask has to monitor your computation.
    - First section with pandas will crash
    - Let's scale up to a big computer; feel free to work through this section on your own, and I'll talk about what each cell is doing as I go.
        - **Stop at cluster options to set the worker size to medium and environment to nebari-demo; these are also options which are configured in the nebari-config.yml, which is the single source of configuration for nebari.**
    - <Lead them through connecting to the gateway, then creating a cluster and a client, at which point they should be able to connect to the dashboard. Ask them to click on a few dashboards and watch computation progress.>
    - Run the array example - use small workers, they launch faster. Set the minimum to 1.
    - Make sure to shut down the cluster when you're done.

# CDS Dashboards

- Short intro — Many of you may have heard of some of the really cool dashboarding tools that exist in the python ecosystem, that allow you to build interactive data visualizations for the web - like Voila, Bokeh, Plotly Dash, Streamlit, and some others. Nebari has the ability to create and host dashboards built with these frameworks, and it allows you to share these dashboards with whoever you want.
- Let’s walk through making a dashboard!
- Share the link to Eskild's dashboards as a way to demonstrate that it's really sharable.

# Setup

- Key takeaways
    1. Quickly initialize config file
    2. Easily deploy cluster
- Motivation
    - *spent hundreds of hours integrating these core services and making it as easy as possible to get started.*
- Setup steps diagram
- **Walkthrough CLI**
    - *Show help commands*
    - *Walk through guided-init*
- **Config file sections**
    - *Highlight several import sections*
- Show the built deployment!
    - *Mention that a deployment will not fall under a “Free Tier”*
        - *InfraCost - used to estimate the base monthly cost*

# Conclusion

- Key takeaways
    1. What is Nebari?
        - *An opinionated JupyterHub distribution (running on Kubernetes), managed by Terraform, yet flexible*
        -
    2. Key features
        - *Manage environments with conda-store*
        - *Scale your python code with Dask*
        - *Share dashboards and visualizations*
    3. Deploy a Nebari cluster
        - *Get started quickly but have the flexibility to make it own*
- *Lean towards Q/A if we’re short on time*