spinTwo
Enabling Scientific Computing

CARLA 2025
LATIN AMERICA HIGH PERFORMANCE COMPUTING CONFERENCE

FROM THE CLI TO THE BROWSER: OPEN ONDEMAND MAKING HPC EASY TO ACCESS

Hugo R Hernández (virtual)
Kingston, Jamaica – September 25, 2025

AGENDA

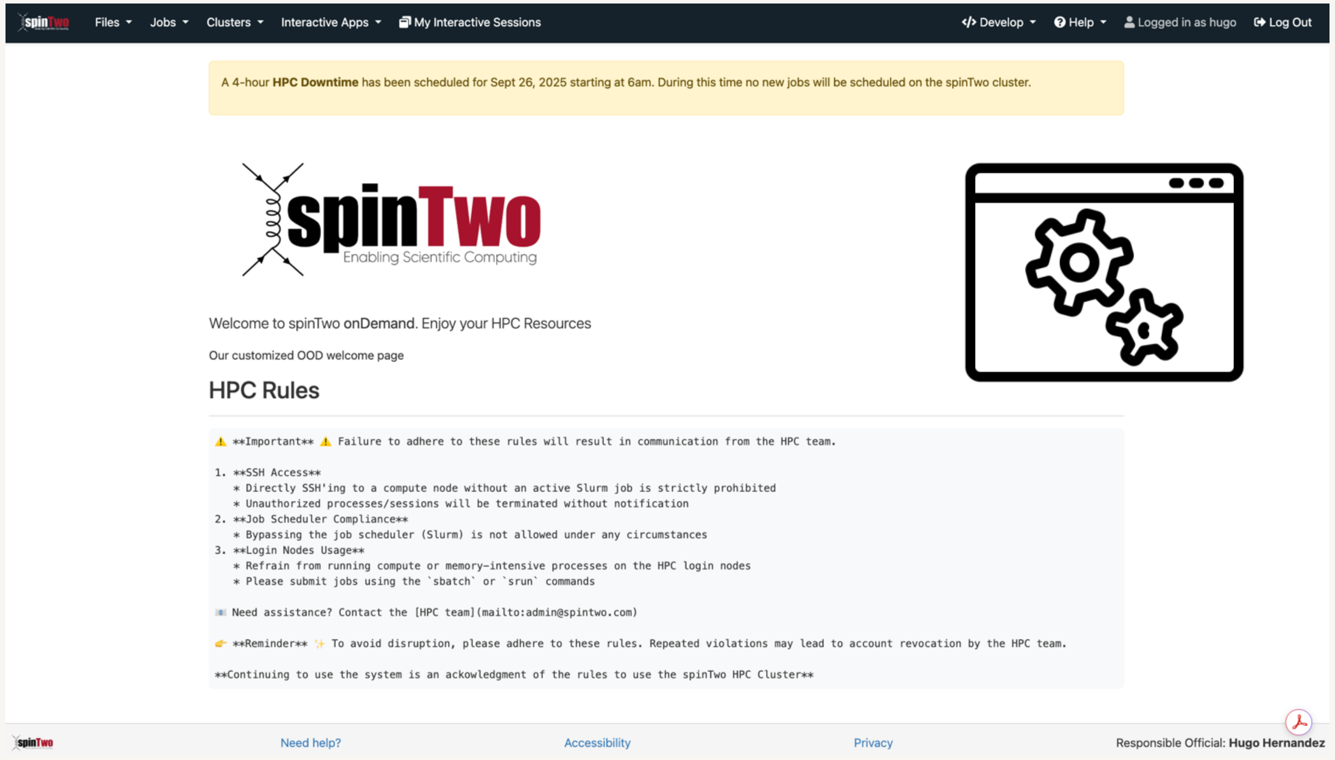
- 🎯 Introduction
- 🎯 Open OnDemand Technical Deep Dive
- 🎯 Deployment, Automation, and Performance
- 🎯 Security and Best Practices
- 🎯 Integration Scenarios
- 🎯 Summary
- 🎯 Q & A

INTRODUCTION: OPEN ONDEMAND

OPEN OnDemand

What is Open OnDemand (OOD)?

- Open-source HPC portal developed by **Ohio Supercomputer Center (OSC)**
- Provides a **web-based interface** to HPC clusters
- **Eliminates need** for CLI-only workflows



A 4-hour **HPC Downtime** has been scheduled for Sept 26, 2025 starting at 6am. During this time no new jobs will be scheduled on the spinTwo cluster.

Welcome to spinTwo **onDemand**. Enjoy your HPC Resources

Our customized OOD welcome page

HPC Rules

```
⚠️ **Important** ⚠️ Failure to adhere to these rules will result in communication from the HPC team.

1. **SSH Access**
   * Directly SSH'ing to a compute node without an active Slurm job is strictly prohibited
   * Unauthorized processes/sessions will be terminated without notification
2. **Job Scheduler Compliance**
   * Bypassing the job scheduler (Slurm) is not allowed under any circumstances
3. **Login Nodes Usage**
   * Refrain from running compute or memory-intensive processes on the HPC login nodes
   * Please submit jobs using the `sbatch` or `srun` commands

📧 Need assistance? Contact the [HPC team](mailto:admin@spintwo.com)

👉 **Reminder** ✨ To avoid disruption, please adhere to these rules. Repeated violations may lead to account revocation by the HPC team.

**Continuing to use the system is an ackowledgment of the rules to use the spinTwo HPC Cluster**
```

Responsible Official: **Hugo Hernandez**

Why is it relevant?

- **Lowers barrier** for HPC adoption (researchers, students, industry)
- Access HPC resources **from any browser, anywhere**
- Supports **interactive apps** (Jupyter, RStudio, MATLAB, VNC desktops)
- Integrates with **Slurm and other schedulers** for job submission
- Streamlines **file management, job monitoring,** and **collaboration**
- Enables **Single Sign-On (SSO) integration** for secure, seamless user access

Key Features

- **Zero installation**: run OOD entirely in your browser
- Easy to use; **start computing immediately**
- Shell access through the browser, **no SSH client needed**
- Run **graphical apps** on the cluster, view them **in the browser**
- **Seamless** data management
- Submit, monitor, and cancel **Slurm jobs**
- Extensive framework for **developers**

INTRODUCTION: WHO ARE WE?

spinTwo delivers cutting-edge optimization **solutions** powered by advanced supercomputing technologies

- U.S.-based HPC and scientific computing experts; **office in Colombia**
- Secure, optimized, and scalable HPC environments
- Combines **scientific domain knowledge** with technical expertise

Focus areas

- HPC cluster design & management
- AI-driven solutions
- Remote desktop and GUI optimization
- Secure data environments
- Workflow optimization for interactive and batch jobs
- HPC security assessments

spinTwo
Enabling Scientific Computing

Your Strategic Partner in HPC Solutions! Build, Optimize, and Accelerate

PARTNER ECOSYSTEM

spinTwo is partnering with leading technology providers to deliver a complete HPC and AI environment, combining seamless access, interactive applications, and scalable workload management.

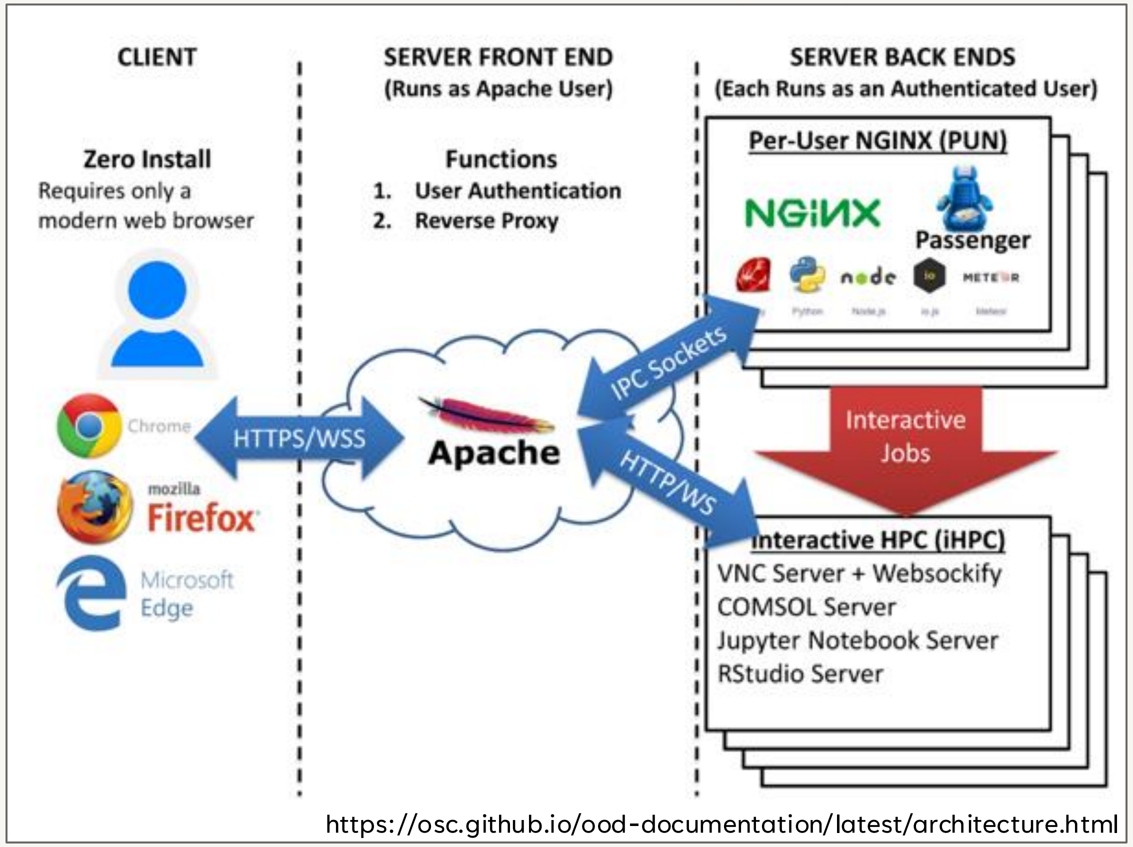
- **Ohio Supercomputing Center – Open OnDemand**: Browser-based HPC portal, job submission, file management, interactive apps
- **Cendio – ThinLinc**: High-performance remote desktop solution, GPU-optimized for interactive apps
- **SchedMD – Slurm & Slinky**: HPC workload manager for batch and interactive jobs, scalable to large clusters, with seamless integration between Kubernetes and HPC environments.
- **VAST Data:** AI-ready, high-performance storage for fast, reliable access to large datasets

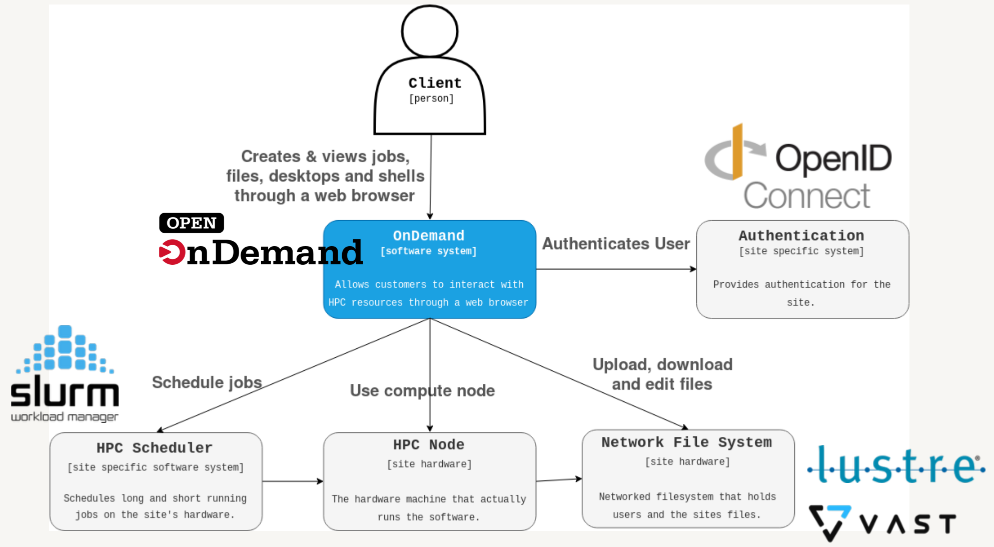
OPEN ONDEMAND ARCHITECTURE

- Apache is the server front end, running as the Apache user, and accepting all requests from users and serves four primary functions:

  1. Authenticates user
  2. Starts Per-User NGINX processes (PUNs)
  3. Reverse proxies each user to her PUN via Unix domain sockets
  4. Reverse proxies to interactive apps running on compute nodes (RStudio, Jupyter, VNC desktop) via TCP sockets

- The Per-User NGINX serves web apps in Ruby and NodeJS and is how users submit jobs and start interactive apps.




Client
[person]
Creates & views jobs, files, desktops and shells through a web browser
OPEN OnDemand
OnDemand
[software system]
Allows customers to interact with HPC resources through a web browser
Authenticates User
OpenID Connect
Authentication
[site specific system]
Provides authentication for the site.
slurm workload manager
Schedule jobs
Use compute node
Upload, download and edit files
HPC Scheduler
[site specific software system]
Schedules long and short running jobs on the site's hardware.
HPC Node
[site hardware]
The hardware machine that actually runs the software.
Network File System
[site hardware]
Networked filesystem that holds users and the sites files.
lustre
VAST

Web Server Layer: Apache/Nginx + Passenger, Ruby on Rails
Batch Scheduler Layer: Slurm, PBS, LSF
Interactive Apps Layer: Jupyter, RStudio, MATLAB, VNC/ThinLinc
Storage Layer: NFS, Lustre, GPFS, Object storage, Globus
Authentication Layer: LDAP, OAuth2, Shibboleth, PAM, SSO via OIDC

DEPLOYING OPEN ONDEMAND VIA ANSIBLE

Open OnDemand **is deployed and configured using Ansible**, which automate the installation process, apply system-wide configurations, and ensure consistent environments across HPC clusters.

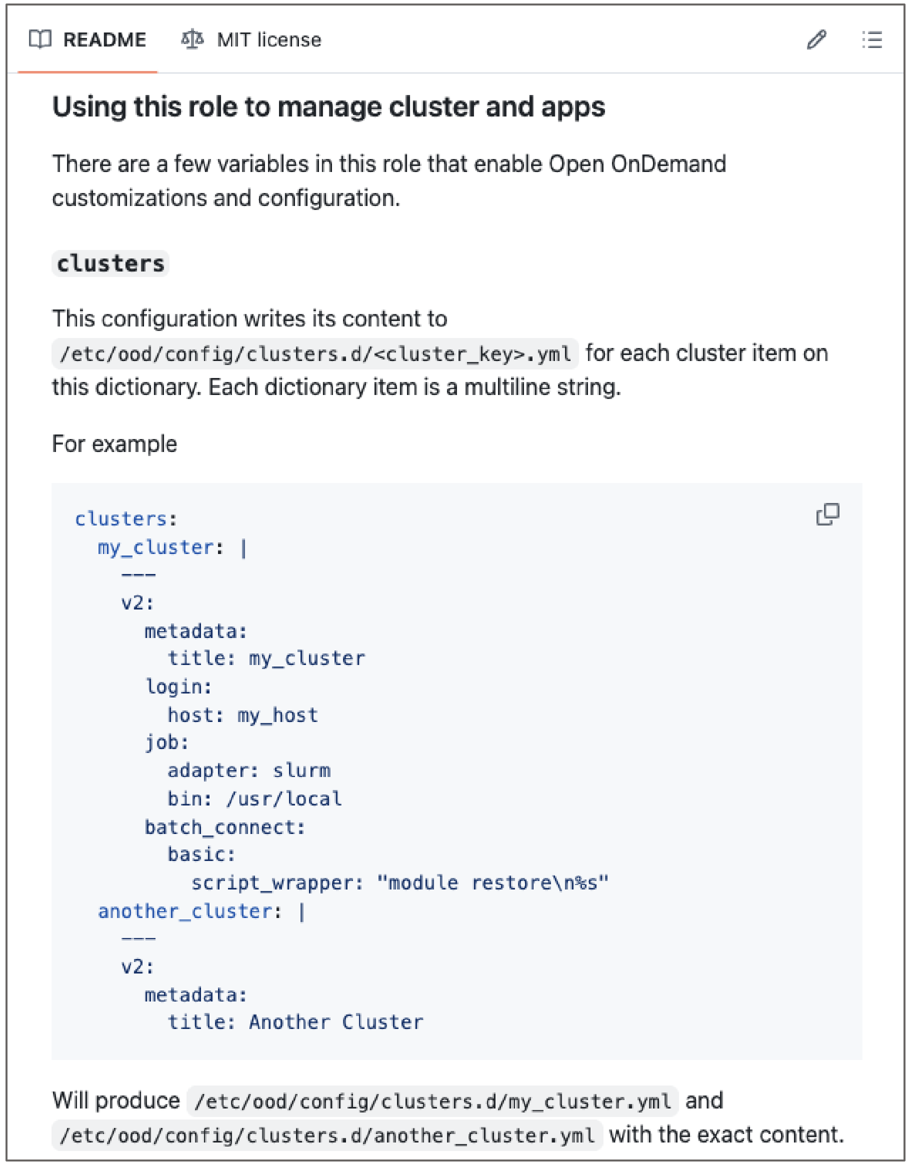
This approach streamlines setup, reduces manual errors, and makes it easier to maintain and update over time.

Repository Structure:

```
├── ood_install.yml
├── inventory.yml
├── overrides.yml
├── vars-common.yml
├── vars-prod.yml
├── roles/
│  └── post-install/
└── README.md
```

- **Automated Deployment:** Ansible playbooks install and configure OOD across OOD server(s). Prepare HPC nodes with required packages
- **Configuration Management:** Centralized control of web server, authentication, storage, and scheduler settings
- **Reproducible & Repeatable:** Ensures consistent OOD environments across clusters, minimizing manual errors
- **Integration Ready:** Easily configure SSO, OIDC, Slurm, and storage systems
- **Scalable Updates:** Apply updates or new modules to all nodes in one step

https://github.com/OSC/ood-ansible



Using this role to manage cluster and apps

There are a few variables in this role that enable Open OnDemand customizations and configuration.

`clusters`

This configuration writes its content to `/etc/ood/config/clusters.d/<cluster_key>.yml` for each cluster item on this dictionary. Each dictionary item is a multiline string.

For example

```
clusters:
  my_cluster: |
    ---
    v2:
      metadata:
        title: my_cluster
      login:
        host: my_host
      job:
        adapter: slurm
        bin: /usr/local
      batch_connect:
        basic:
          script_wrapper: "module restore\n%s"
  another_cluster: |
    ---
    v2:
      metadata:
        title: Another Cluster
```

Will produce `/etc/ood/config/clusters.d/my_cluster.yml` and `/etc/ood/config/clusters.d/another_cluster.yml` with the exact content.

PLAYBOOK BREAKDOWN

Role Installation

To install the official OOD Ansible role from Ansible Galaxy:

```
ansible-galaxy role install osc.open_ondemand
```

This installs the role into your Ansible roles path (default: `~/.ansible/roles/`)

Playbook Breakdown

The playbook `install_ood.yml` runs two roles:

```
- name: Deploy OOD
  hosts: ood-head
  become: true
  vars_files:
    - vars-common.yml
  roles:
    - role: osc.open_ondemand
    - role: ood_post_install
```

The `osc.open_ondemand` role installs and configures Open OnDemand. This includes:

- Package installation
- Apache and Passenger setup
- OOD portal configuration

The `ood_post_install` role handles custom modifications after the base OOD installation, such as:

- Adding site-specific branding & logos
- Configuring user mapping
- Customizing the dashboard & interactive apps
- Any additional site-wide tweaks

Running the Playbook

```
ansible-playbook -i inventory.yaml ood_install.yml \
                 -J --extra-vars=@overrides.yml \
                    --extra-vars=@ood-secrets
```

where the file `overrides.yml` provides the values to customize the installation and `ood-secrets.yml` is a Vault encrypted file containing OIDC secrets.

AUTHENTICATION & SINGLE SIGN-ON

Open OnDemand supports secure, browser-based login via **Single Sign-On (SSO)** using **OpenID Connect (OIDC)**.
This allows users to authenticate with:

- University SSO providers for federated login (e.g., Shibboleth/SAML, AzureAD)
- **LDAP / Active Directory integration** – centralized user management for traditional authentication
- Google accounts (with domain restriction)
- Federated logins via a broker like Keycloak

The OIDC configuration is defined in `/etc/ood/config/ood_portal.yml`. The main parameters are:

- **oidc_uri:** `/oidc` – route authentication requests/responses and mount OIDC endpoint
- **oidc_provider_metadata_url:** Discovery URL for IdP configuration (e.g., authorization endpoint)
- **oidc_client_id & oidc_client_secret:** Registers OOD as a client with your IdP (Keycloak, Azure AD, Google, Okta, etc.)
- **oidc_remote_user_claim:** Defines which claim from the ID token (e.g., preferred_username, email) maps to `REMOTE_USER` in OOD

Create and encrypt `ood-secrets.yml`

You can create the encrypted file outright using :

```
ansible-vault create ood-secrets.yml
```

or alternatively create a text file, then encrypt it:

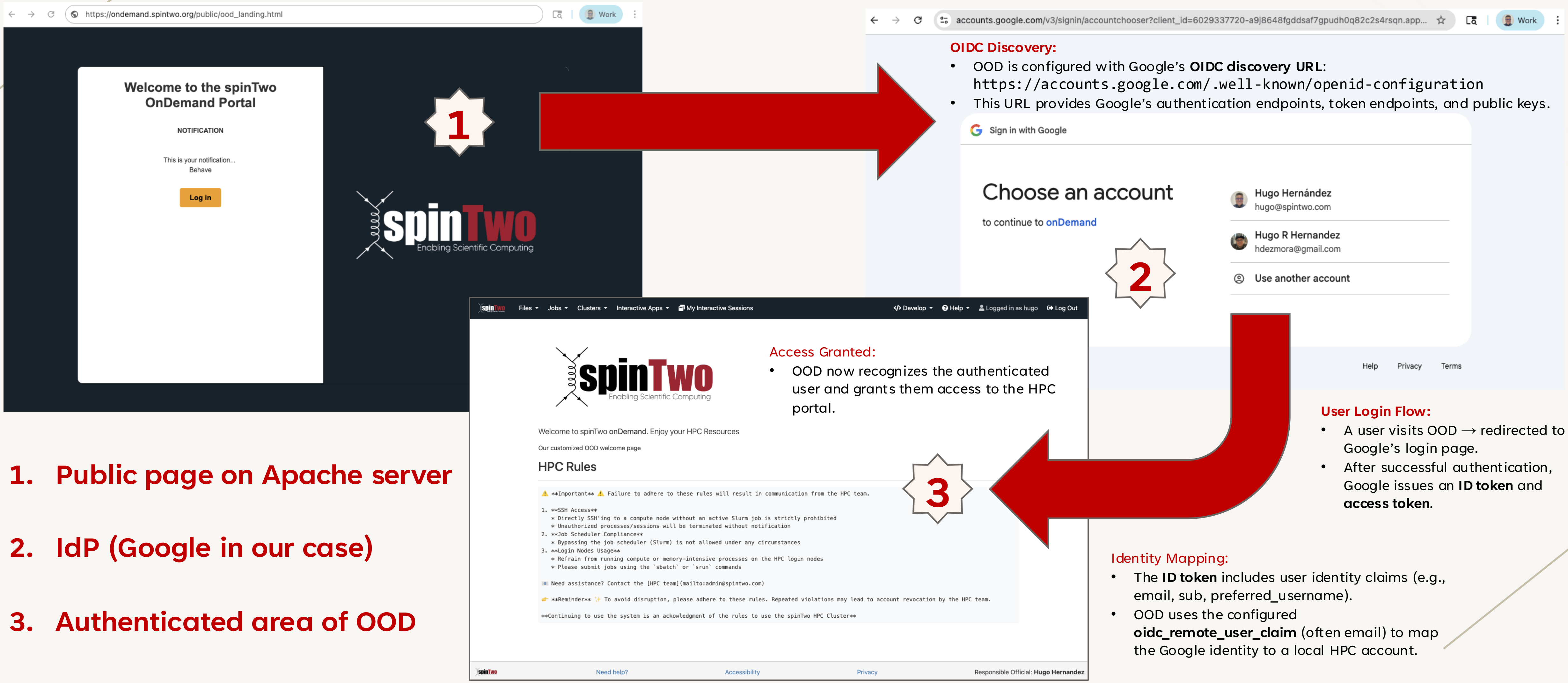
```
ansible-vault encrypt ood-secrets.yml
```

```
vault_oidc_client_id: <OIDC_CLIENT_ID>
vault_oidc_client_secret: <OIDC_CLIENT_SECRET>
vault_ood_oidc_crypto_passphrase: <OOD_OIDC_CRYPTO_PASSPHRASE>
```

```
### values associated to the creation of `/etc/ood/config/ood_portal.yaml`
httpd_port: 443

# Use OIDC for authentification
httpd_auth:
  - "AuthType openid-connect"
  - "Require valid-user"

# OIDC Parameters
oidc_uri: "/oidc"
oidc_provider_metadata_url: "https://accounts.google.com/.well-known/openid-configuration"
oidc_client_id: "{{ vault_oidc_client_id }}"
oidc_client_secret: "{{ vault_oidc_client_secret }}"
oidc_remote_user_claim: email # spinTwoOrgUserId << if your org has in place its own SSO method
oidc_scope: "openid profile email"
oidc_session_inactivity_timeout: 28800
oidc_session_max_duration: 28800
oidc_state_max_number_of_cookies: "10 true"
```

AUTHENTICATION & SINGLE SIGN-ON (CONT.)

https://ondemand.spintwo.org/


https://ondemand.spintwo.org/public/ood_landing.html
Work
Welcome to the spinTwo OnDemand Portal
NOTIFICATION
This is your notification... Behave
Log in
spinTwo Enabling Scientific Computing
1
accounts.google.com/v3/signin/accountchooser?client_id=6029337720-a9j8648fgddsaf7gpudh0q82c2s4rsqn.app...
Sign in with Google
Choose an account
to continue to onDemand
Hugo Hernández
hugo@spintwo.com
Hugo R Hernandez
hdezmora@gmail.com
Use another account
Help
Privacy
Terms
2
spinTwo
Files
Jobs
Clusters
Interactive Apps
My Interactive Sessions
Develop
Help
Logged in as hugo
Log Out
Welcome to spinTwo onDemand. Enjoy your HPC Resources
Our customized OOD welcome page
3
Need help?
Accessibility
Privacy
Responsible Official: Hugo Hernandez

HPC Rules

```
⚠️ **Important** ⚠️ Failure to adhere to these rules will result in communication from the HPC team.

1. **SSH Access**
   * Directly SSH'ing to a compute node without an active Slurm job is strictly prohibited
   * Unauthorized processes/sessions will be terminated without notification
2. **Job Scheduler Compliance**
   * Bypassing the job scheduler (Slurm) is not allowed under any circumstances
3. **Login Nodes Usage**
   * Refrain from running compute or memory-intensive processes on the HPC login nodes
   * Please submit jobs using the `sbatch` or `srun` commands

📧 Need assistance? Contact the [HPC team](mailto:admin@spintwo.com)

👉 **Reminder** ✨ To avoid disruption, please adhere to these rules. Repeated violations may lead to account revocation by the HPC team.

**Continuing to use the system is an ackowledgment of the rules to use the spinTwo HPC Cluster**
```

OIDC Discovery:
- OOD is configured with Google's **OIDC discovery URL**: `https://accounts.google.com/.well-known/openid-configuration`
- This URL provides Google's authentication endpoints, token endpoints, and public keys.

Access Granted:
- OOD now recognizes the authenticated user and grants them access to the HPC portal.

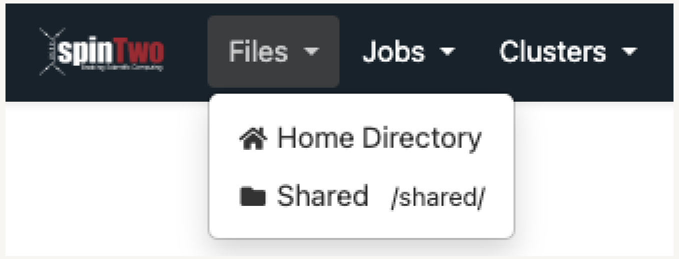
User Login Flow:
- A user visits OOD → redirected to Google's login page.
- After successful authentication, Google issues an **ID token** and **access token.**

Identity Mapping:
- The **ID token** includes user identity claims (e.g., email, sub, preferred_username).
- OOD uses the configured **oidc_remote_user_claim** (often email) to map the Google identity to a local HPC account.

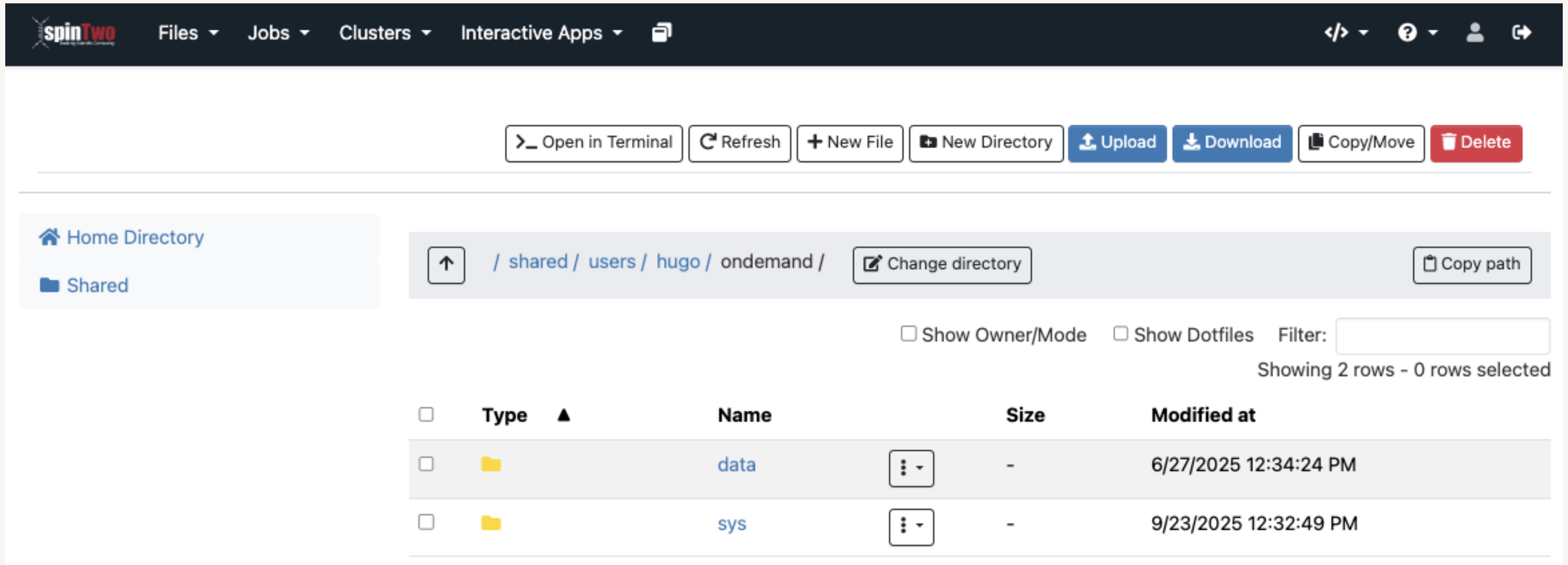
1. **Public page on Apache server**
2. **IdP (Google in our case)**
3. **Authenticated area of OOD**

FILE MANAGEMENT AND STORAGE

Web-Based File Management: Browse, upload, download, and edit files on HPC storage directly from your browser



Note: OOD's File app has no built-in mechanism to show recall progress from archiving systems like tapes. This could confuse users unless documented.



Type	Name	Size	Modified at
📁	data	-	6/27/2025 12:34:24 PM
📁	sys	-	9/23/2025 12:32:49 PM

Caution: staging delays can affect UX when accessing offline data!

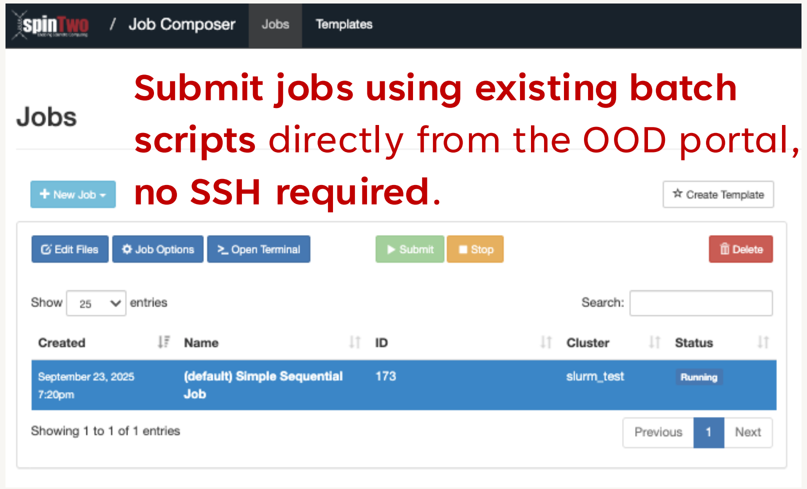
- **Integration with HPC Filesystems:** Supports NFS, Lustre, GPFS, and other mounted storage
- **No Separate PUN Required:** Runs through the main OOD portal, unlike interactive apps
- **Secure Access:** Respects user permissions and integrates with LDAP, OIDC, or other authentication methods
- **Convenience for Researchers:** Enables quick access to data without SSH or command-line tools

ood_portal.yml

```
# Add these filesytems to the Dashboard
filesystems:
  - name: Lustre Filesystem
    path: /shared/users/#{User.new.name}
  - name: VAST Shares
    path: /vast/#{User.new.name}
```

Runs within the main OOD web server:
Apache/Nginx + Passenger + Ruby on Rails

SUBMITTING AND MONITORING BATCH JOBS



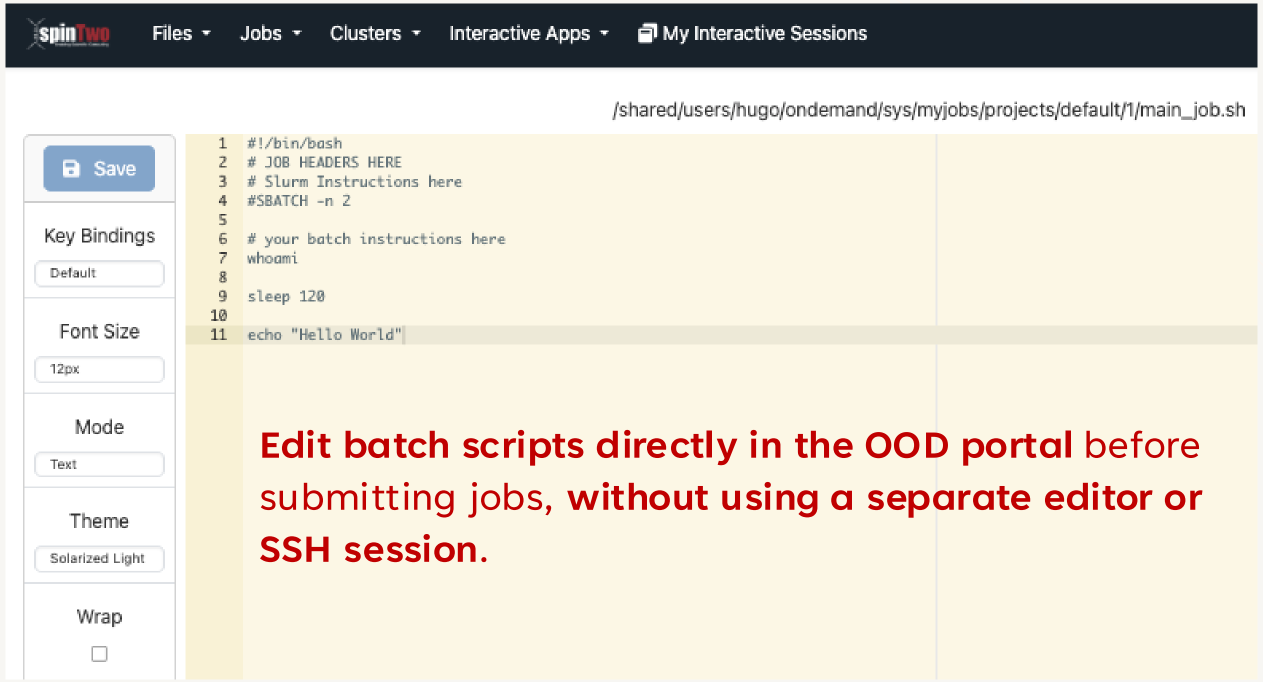
Jobs

Submit jobs using existing batch scripts directly from the OOD portal, **no SSH required.**

Show 25 entries   Search:

Created	Name	ID	Cluster	Status
September 23, 2025 7:20pm	(default) Simple Sequential Job	173	slurm_test	Running

Showing 1 to 1 of 1 entries

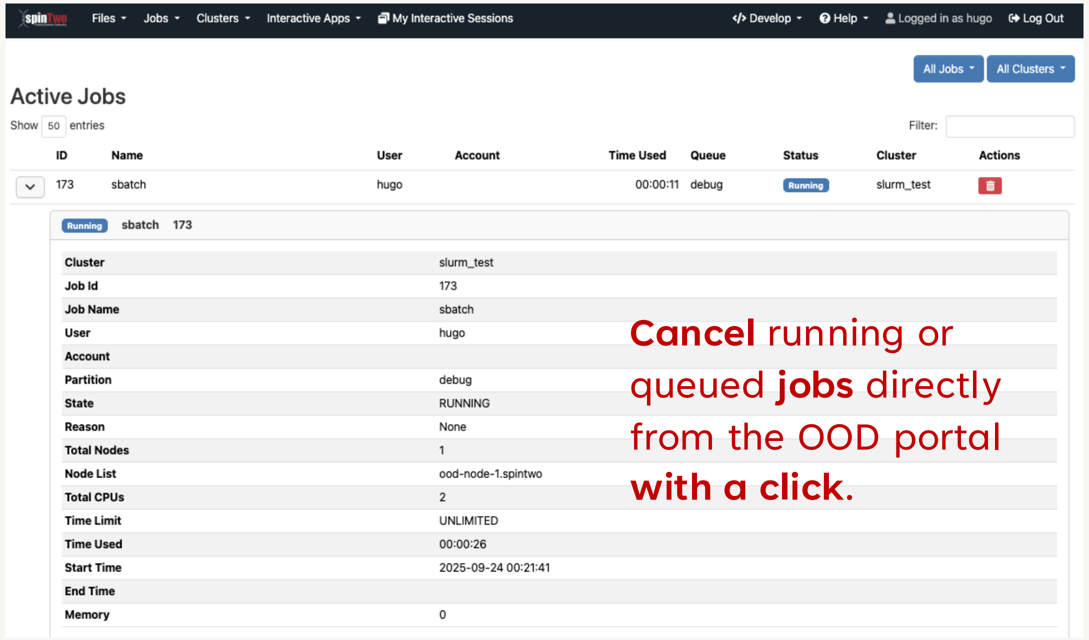


/shared/users/hugo/ondemand/sys/myjobs/projects/default/1/main_job.sh

```
#!/bin/bash
# JOB HEADERS HERE
# Slurm Instructions here
#SBATCH -n 2

# your batch instructions here
whoami

sleep 120

echo "Hello World"
```

Key Bindings	Default
Font Size	12px
Mode	Text
Theme	Solarized Light
Wrap	☐

Edit batch scripts directly in the OOD portal before submitting jobs, **without using a separate editor or SSH session.**



Active Jobs

Show 50 entries   Filter:

ID	Name	User	Account	Time Used	Queue	Status	Cluster	Actions
173	sbatch	hugo		00:00:11	debug	Running	slurm_test	

Running sbatch 173

Cluster	slurm_test
Job Id	173
Job Name	sbatch
User	hugo
Account	
Partition	debug
State	RUNNING
Reason	None
Total Nodes	1
Node List	ood-node-1.spintwo
Total CPUs	2
Time Limit	UNLIMITED
Time Used	00:00:26
Start Time	2025-09-24 00:21:41
End Time	
Memory	0

Cancel running or queued **jobs** directly from the OOD portal **with a click.**

- **Web-Based Job Management:** Submit, monitor, and manage HPC jobs via a browser interface
- **Supports Multiple Schedulers:** Slurm, PBS, LSF
- **Job Templates & Custom Scripts:** Easily create and reuse batch scripts for common workflows
- **Status & Monitoring:** Visualize job states, logs, and resource usage

SLURM CONFIGURATION

```
[root@ood-head clusters.d]# cat slurm.yml
---
v2:
  metadata:
    title: slurm_test
  login:
    host: hpc.spintwo.org
  job:
    adapter: slurm
    bin: /hpc/apps/slurm/default/bin/
  batch_connect:
    min_port: 30000
    max_port: 30999
    basic:
      script_wrapper: |
        module purge
        %s
    vnc:
      script_wrapper: |
        module purge
        export PATH="/opt/TurboVNC/bin:$PATH"
        export WEBSOCKIFY_CMD="/usr/bin/websockify"
        %s
```

Key Points

- `v2:` ensures that OOD interprets the YAML according to the newer Batch Connect syntax and features
- `metadata:` human-friendly info for the portal (title, description, etc.)
- `host:` the login node or head node for Slurm
- `job:` defines how OOD interacts with the job scheduler
- `batch_connect:` configures interactive session options (like basic shell or VNC) and TCP port ranges
- `basic:` defines a simple interactive session type for the Batch Connect app
- `script_wrapper:` defines the shell commands that wrap around the user’s submitted commands (`%s` is replaced by the user’s commands)

MULTI-CLUSTERS

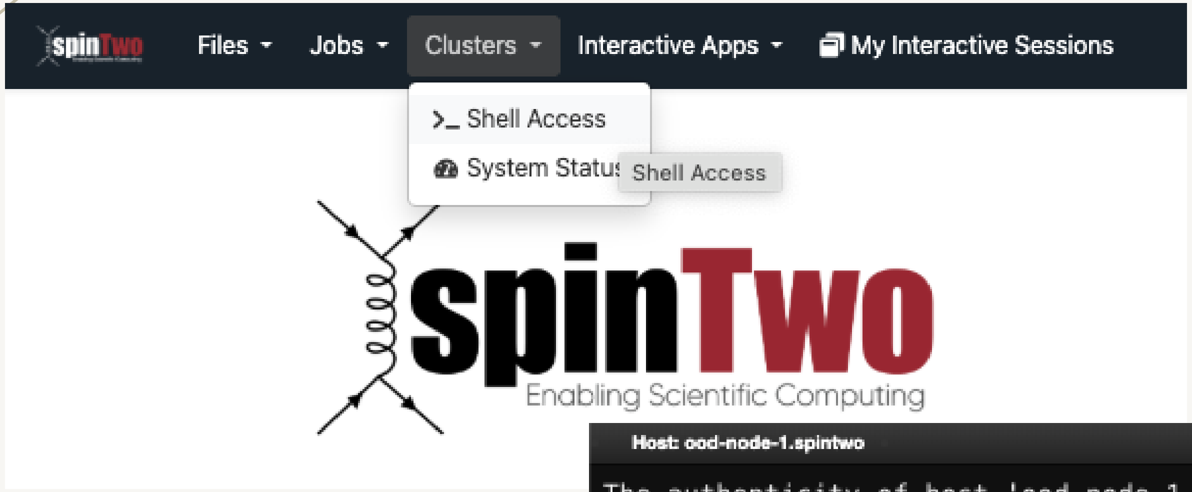
```
[root@ood-head ~]# cat /etc/ood/config/clusters.d/slurm.yml
clusters:
  dirac: |
    v2:
      metadata:
        title: dirac
      login:
        host: dirac.spintwo.org
      job:
        adapter: slurm
        bin: /hpc/apps/slurm/default/bin/
        cluster: dirac
      batch_connect:
        min_port: 30000
        max_port: 30999
        basic:
          script_wrapper: |
            module purge
            module python/3.11
            module cuda/12.1
            %s
        vnc:
          script_wrapper: |
            module purge
            export PATH="/opt/TurboVNC/bin:$PATH"
            export WEBSOCKIFY_CMD="/usr/bin/websockify"
            %s
```

```
#
# Second cluster
 feynman: |
    v2:
      metadata:
        title: feynman
      login:
        host: feynman.spintwo.org
      job:
        adapter: slurm
        bin: /hpc/apps/slurm/default/bin/
        cluster: feynman
      batch_connect:
        min_port: 30000
        max_port: 90999
        basic:
          script_wrapper: |
            module purge
            module gcc/12
            module openmpi/4.1.5
            %s
        vnc:
          script_wrapper: |
            module purge
            export PATH="/opt/TurboVNC/bin:$PATH"
            export WEBSOCKIFY_CMD="/usr/bin/websockify"
            %s
```

VIRTUAL SHELL

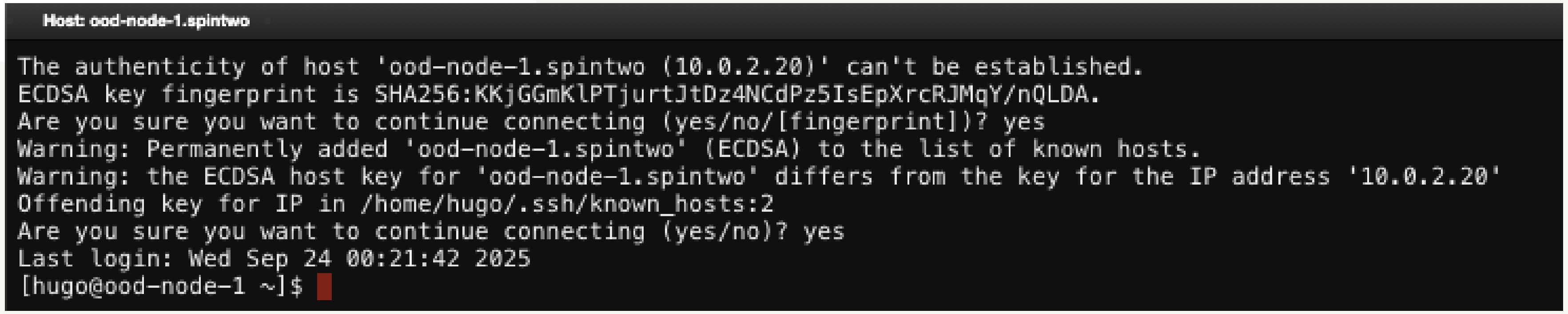
vars-prod.yml

```
node_glob: "node000[1-9].spinTwo.org:node001[0-9].spintwo.org:node0020.spintwo.org”
ood_apps:
  shell:
    env:
      # restrict ssh access to this nodes
      ood_sshhost_allowlist: "{{ blade_glob }}:{{ node_glob }}:{{ rin_glob }}:{{ int_glob }}"
```



The SSH access to compute nodes is controlled by two env vars in `/etc/ood/config/apps/shell/env`:

- `OOD_DEFAULT_SSHHOST`: this is the node to SSH from the clusters->shell access button. This can also be set in the cluster configuration under login. This can be a load-balanced login node.
- `OOD_SSHHOST_ALLOWLIST`: list of nodes to which SSH is allowed from a running job. This is where you'd land if you click on the node name of your interactive session. This should be colon separated list of GLOBs.



```
Host: ood-node-1.spintwo
The authenticity of host 'ood-node-1.spintwo (10.0.2.20)' can't be established.
ECDSA key fingerprint is SHA256:KKjGGmKlPTjurtJtDz4NCdPz5IsEpXrcRJMqY/nQLDA.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ood-node-1.spintwo' (ECDSA) to the list of known hosts.
Warning: the ECDSA host key for 'ood-node-1.spintwo' differs from the key for the IP address '10.0.2.20'
Offending key for IP in /home/hugo/.ssh/known_hosts:2
Are you sure you want to continue connecting (yes/no)? yes
Last login: Wed Sep 24 00:21:42 2025
[hugo@ood-node-1 ~]$
```

OOD provides users with access into a virtual shell, a web-based terminal interface to the HPC clusters. It allows secure, browser-based command-line access to login or compute nodes without needing a separate SSH client.

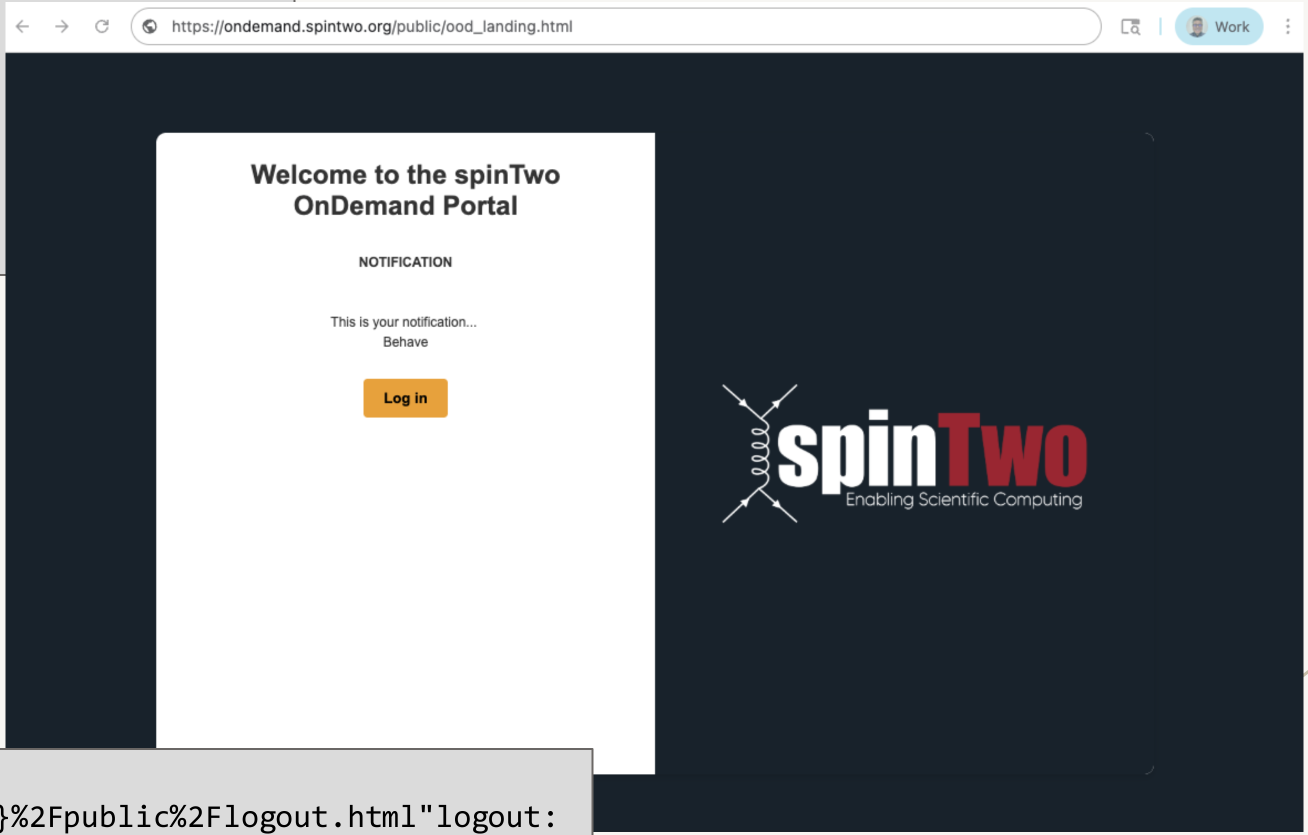
This feature enables users to manage files, run commands, monitor jobs, and interact with the system remotely and conveniently through the OOD portal.

PORTAL CUSTOMIZATION

Landing Page

vars-common.yml

```
# Set the landing page
root_uri: "{{ '/public/ood_landing.html' if landing.create else none }}"
landing:
  create: true
  title: "Welcome to the spinTwo OnDemand Portal"
  logo: "{{ logos.dashboard | basename }}"
  login_button: "Login"
  notification: |
    <strong><center>NOTIFICATION</center></strong><br>
    This is your notification... <br>Behave
```


https://ondemand.spintwo.org/public/ood_landing.html
Work
Welcome to the spinTwo OnDemand Portal
NOTIFICATION
This is your notification...
Behave
Log in
spinTwo
Enabling Scientific Computing

The Open OnDemand landing page **welcomes users** to the spinTwo portal with a **custom title, logo, and notification banner**. From here, users authenticate using **the SSO login button**, ensuring a simple and consistent entry point to HPC resources.

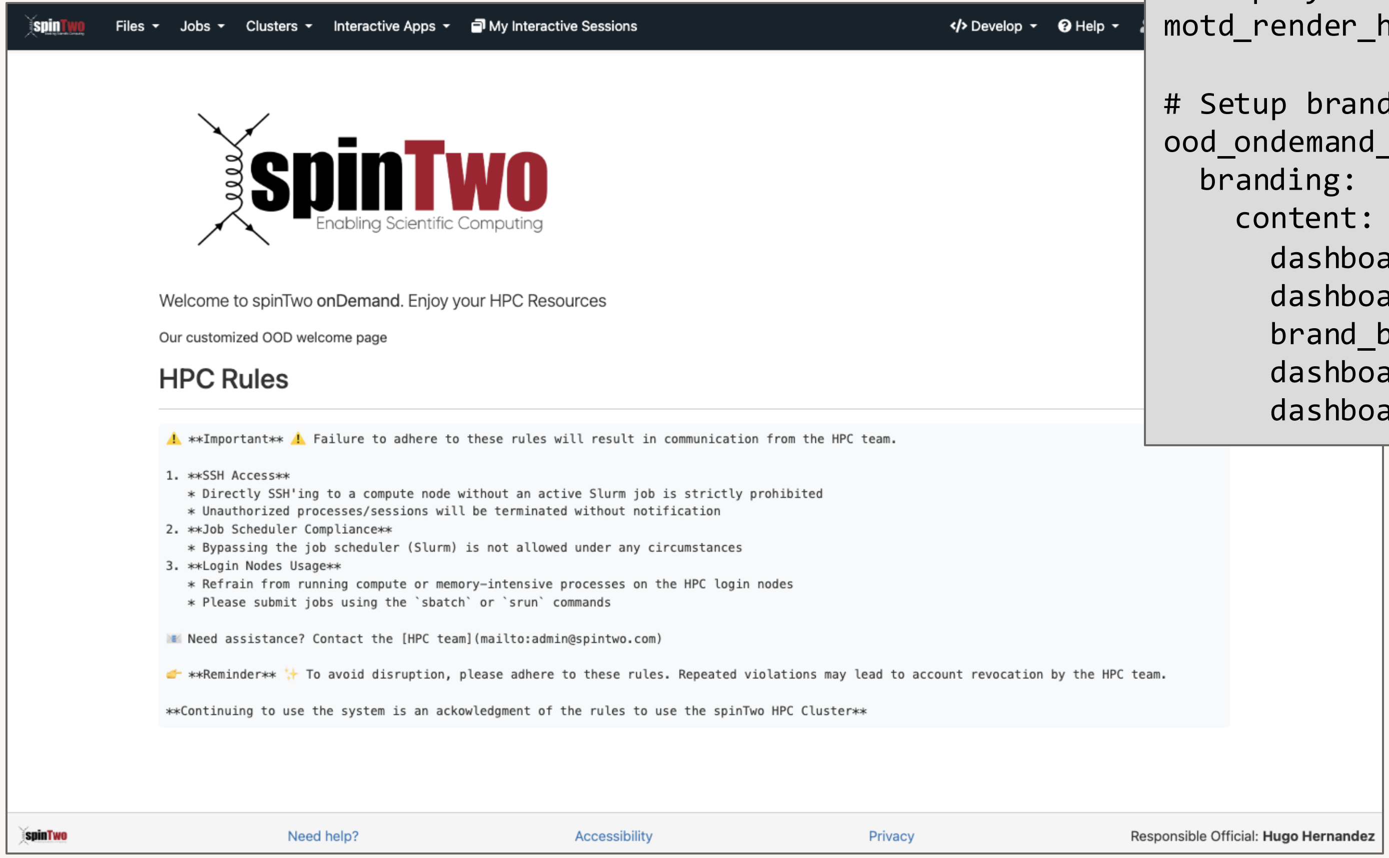
```
# set logout; redirect_uri must be urlencoded
logout_redirect: "/oidc?logout=https%3A%2F%2F{{ servername }}%2Fpublic%2Flogout.html"logout:
  logo: "{{ logos.dashboard | basename }
```

PORTAL CUSTOMIZATION (CONT.)

Welcome Page

vars-common.yml

```
custom_welcome_html: |
  %{logo_img_tag}
  <p class="lead"> Welcome to spinTwo <b>onDemand</b>. Enjoy your HPC
Resources</p>
  <p>Our customized OOD welcome page</p>

# Display Message of the Day in /etc/motd-ood
motd_render_html: true

# Setup branding
ood_ondemand_d_configs:
  branding:
    content:
      dashboard_title: spinTwo OnDemand
      dashboard_logo: "/public/{{ logos.dashboard | basename }}"
      brand_bg_color: "{{ branding_color }}"
      dashboard_logo_height: 250
      dashboard_header_img_logo: "/public/{{ logos.header | basename }}"
```



Welcome to spinTwo **onDemand**. Enjoy your HPC Resources

Our customized OOD welcome page

HPC Rules

```
⚠️ **Important** ⚠️ Failure to adhere to these rules will result in communication from the HPC team.

1. **SSH Access**
   * Directly SSH'ing to a compute node without an active Slurm job is strictly prohibited
   * Unauthorized processes/sessions will be terminated without notification
2. **Job Scheduler Compliance**
   * Bypassing the job scheduler (Slurm) is not allowed under any circumstances
3. **Login Nodes Usage**
   * Refrain from running compute or memory-intensive processes on the HPC login nodes
   * Please submit jobs using the `sbatch` or `srun` commands

📧 Need assistance? Contact the [HPC team](mailto:admin@spintwo.com)

👉 **Reminder** ✨ To avoid disruption, please adhere to these rules. Repeated violations may lead to account revocation by the HPC team.

**Continuing to use the system is an ackowledgment of the rules to use the spinTwo HPC Cluster**
```

The Welcome Page provides a **customized greeting** with the **portal logos**, a welcome message, and a brief description. It can also display the **Message of the Day (MOTD)** to share important announcements or updates with users (in this case the **HPC Rules**).

PORTAL CUSTOMIZATION (CONT.)

Help and Footer

vars-common.yml

```
footer:
  org_link: "https://spintwo.com"
  logo: /public/{{ logos.dashboard | basename }}
  logo_alt: spinTwo OnDemand
  links:
    - href: "mailto:hpc-admin@spintwo.com"
      text: "Need help?"
    - href: "https://www.spintwo.com/general/accessibility"
      text: "Accessibility"
    - href: "https://www.spintwo.com/privacy"
      text: "Privacy"
  info:
    - "Responsible Official: <strong>Hugo Hernandez</strong>”

# Define the help menu
help_menu:
  - group: "Internal Pages"
  - title: "Accessibility"
    icon: "fas://book"
    url: "https://spintwo.com/accessibility/"
    new_tab: true
  - title: "Privacy"
    icon: "fas://window-restore"
    url: "https://spintwo.com/privacy/"
    new_tab: true
```

The Help menu and Footer provide users with **quick access to documentation, support, and important portal links**, ensuring guidance and resources are always available from any page.

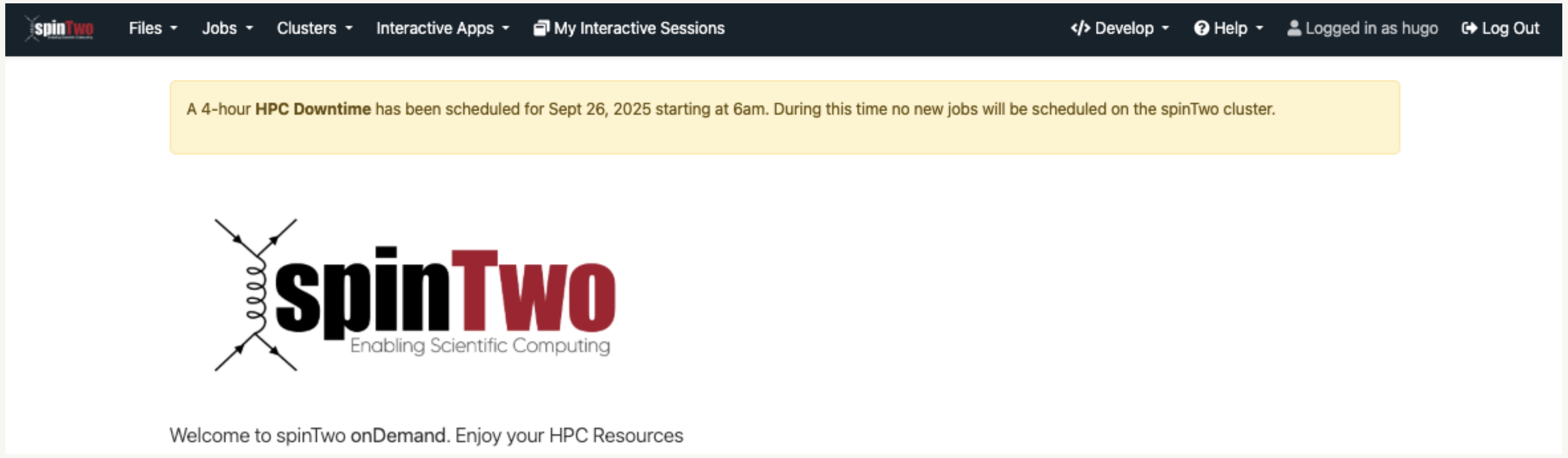
Slurm Script Generator

```
- group: "HPC Tools"
 - page: slurm-script-generator
   icon: "fas://book"
   title: "Slurm Script Generator"
   new_tab: true

custom:
   content:
     custom_pages:
       slurm-script-generator:
         rows:
           - columns:
               - width: 12
                 widgets:
                   - slurm-script-generator
```

PORTAL CUSTOMIZATION (CONT.)

Announcements

Let administrators **share important messages, updates, or alerts** directly in the portal, **keeping HPC users informed** and **aligned with policies and schedules**.



A 4-hour **HPC Downtime** has been scheduled for Sept 26, 2025 starting at 6am. During this time no new jobs will be scheduled on the spinTwo cluster.

Welcome to spinTwo **onDemand**. Enjoy your HPC Resources

Announcements are stored in `/etc/ood/config/announcements.d/`

```
type: warning
dismissible: false
msg: |
  <% if Time.now < Time.new(2025, 9, 26, 12, 0, 0) %>
  A 4-hour **HPC Downtime** has been scheduled for Sept
26, 2025 starting at 6am. During this time no new jobs
will be scheduled on the spinTwo cluster.
  <% end %>
```

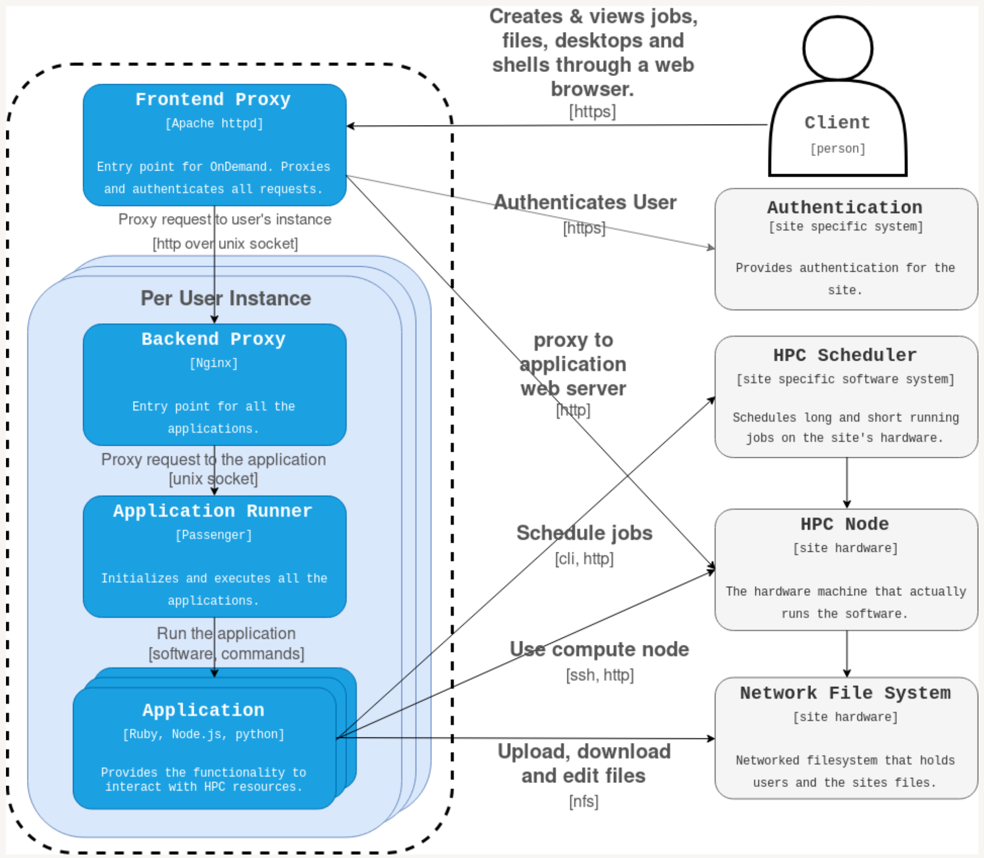
These are managed manually by sysadmins and are **not controlled through Ansible automation**.

The announcement message about the 4-hour HPC downtime will **only be displayed if this condition is true**, i.e., before the specified cutoff time.

PER-USER NGINX (PUN)

A **PUN (Per-User NGINX)** in **Open OnDemand** is a dedicated, lightweight NGINX web server process that runs for each logged-in user. A **PUN is the bridge between OOD's main web portal and the user's HPC environment**

- **User-Specific Web Server:** When a user logs in, OOD starts an NGINX instance that runs under *that user's account*.
- **Security & Permissions:** Because the PUN inherits the user's Unix identity, it enforces the same file permissions and quotas the user has on the HPC system.
- **Session Manager:** The PUN serves the user's web sessions (e.g., launching Jupyter, RStudio, or VNC desktops).
- **Isolation:** Each user's PUN is independent—so one user's applications or crashes don't affect another's.
- **Scalability:** This design allows hundreds or thousands of users to run interactive apps simultaneously without interfering with each other.


Creates & views jobs, files, desktops and shells through a web browser.
[https]
Client
[person]
Frontend Proxy
[Apache httpd]
Entry point for OnDemand. Proxies and authenticates all requests.
Authenticates User
[https]
Authentication
[site specific system]
Provides authentication for the site.
Proxy request to user's instance
[http over unix socket]
Per User Instance
Backend Proxy
[Nginx]
Entry point for all the applications.
proxy to application web server
[http]
HPC Scheduler
[site specific software system]
Schedules long and short running jobs on the site's hardware.
Proxy request to the application
[unix socket]
Application Runner
[Passenger]
Initializes and executes all the applications.
Schedule jobs
[cli, http]
HPC Node
[site hardware]
The hardware machine that actually runs the software.
Run the application
[software commands]
Use compute node
[ssh, http]
Application
[Ruby, Node.js, python]
Provides the functionality to interact with HPC resources.
Upload, download and edit files
[nfs]
Network File System
[site hardware]
Networked filesystem that holds users and the sites files.

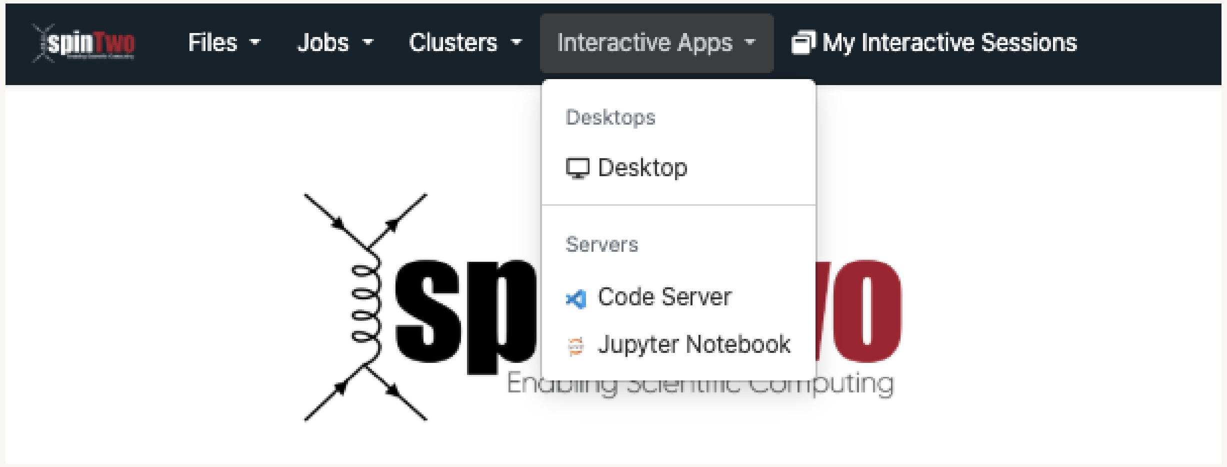
https://osc.github.io/ood-documentation/latest/architecture.html

The Front-end proxy is the only component that is shared with all clients. The **Front-end proxy will create PUN processes** (light blue boxes labeled "Per User Instance").

INTEGRATION SCENARIOS

Interactive Apps

A menu for launching graphical applications (e.g., Jupyter, RStudio, remote desktop) on compute nodes through the job scheduler.

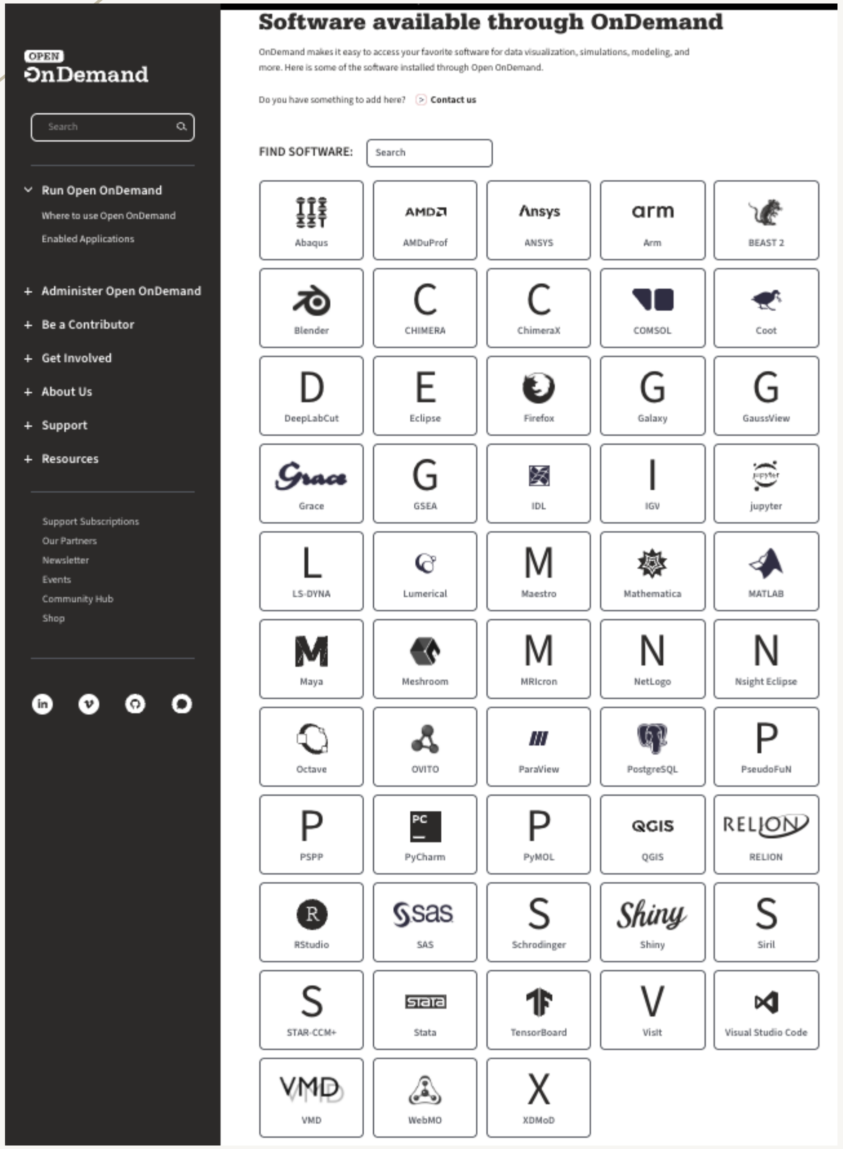


Key Points

- **Launch Applications from Browser:** Run Jupyter, RStudio, MATLAB, VNC/ThinLinc without SSH
- **Per-User NGINX (PUN) Processes:** Each session runs in an isolated environment for security and resource separation
- **Port Allocation:** Dynamic TCP ports assigned for each session automatically
- **Seamless HPC Integration:** Connects directly to cluster nodes and batch schedulers
- **Containers**: Containerized apps for reproducibility (Singularity/Docker)

ENABLED APPS

https://openondemand.org/run-open-ondemand#enabled-applications



OPEN OnDemand

Search

- Run Open OnDemand
  - Where to use Open OnDemand
  - Enabled Applications
- Administer Open OnDemand
- Be a Contributor
- Get Involved
- About Us
- Support
- Resources

Support Subscriptions
Our Partners
Newsletter
Events
Community Hub
Shop

Software available through OnDemand

OnDemand makes it easy to access your favorite software for data visualization, simulations, modeling, and more. Here is some of the software installed through Open OnDemand.

Do you have something to add here? **Contact us**

FIND SOFTWARE: Search

Abaqus	AMDuProf	ANSYS	Arm	BEAST 2
Blender	CHIMERA	ChimeraX	COMSOL	Coot
DeepLabCut	Eclipse	Firefox	Galaxy	GaussView
Grace	GSEA	IDL	IGV	jupyter
LS-DYNA	Lumerical	Maestro	Mathematica	MATLAB
Maya	Meshroom	MRIcron	NetLogo	Nsight Eclipse
Octave	OVITO	ParaView	PostgreSQL	PseudoFuN
PSPP	PyCharm	PyMOL	QGIS	RELION
RStudio	SAS	Schrodinger	Shiny	Siril
STAR-CCM+	Stata	TensorBoard	VisIt	Visual Studio Code
VMD	WebMO	XDMoD		

Open OnDemand supports a wide range of enabled applications—everything from data visualization and modeling tools to scientific and domain-specific software. These are pre-installed or made available through the portal so users can launch them directly without manual setup or installation.

Considerations

- **Dependencies & Modules** – ensuring required software libraries and environment modules are available and compatible
- **HPC Integration** – mapping app launchers to the scheduler (Slurm, PBS, etc.) and compute nodes
- **GPU/CPU Requirements** – properly configuring resources and matching host capabilities
- **Licensing & Access Control** – handling commercial software licenses and user restrictions
- **User Experience** – customizing launch forms and ensuring apps open seamlessly in a web browser

DEFINING YOUR OWN APPS

Defining Applications

Open OnDemand allows administrators to define custom interactive apps—domain-specific tools—by creating simple YAML and script templates. This makes it easy to integrate new software into the portal and provide users with consistent, browser-based access.

- `form.yml`: defines html form for application launch
- `manifest.yml`: Application metadata
- `submit.yml.erb`: Scheduler job submission script
- `templates`
  - `before.sh.erb`: Runs before the main (submission) script. Set passwords, environmental variables, etc
  - `script.sh.erb`: Main script launched by the scheduler
- `completed.md.erb`: Optional. Defines additional information presented in the session card

Create OOD app structure

```
my_app/
├── form.yml
├── manifest.yml
├── submit.yml.erb
├── template
│   ├── before.sh.erb
│   └── script.sh.erb
└── completed.{md,html}.erb
```

Hint: Developers can prepare and share custom apps in Open OnDemand by packaging job templates, forms, and scripts, enabling users to launch applications directly from the portal without manual configuration.

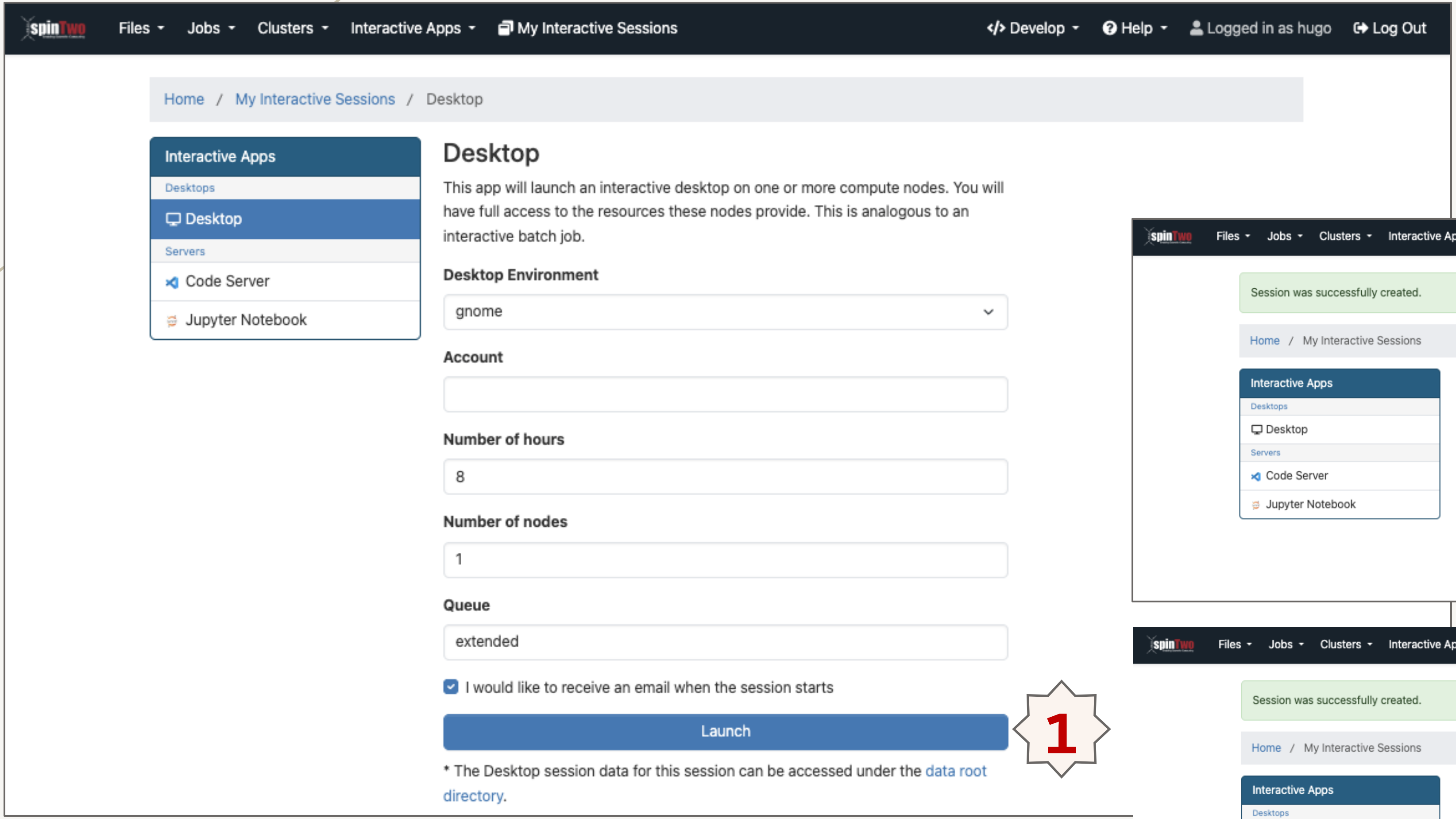
```
# Enable the developer sandbox for these users
developers:
  - eduardo
  - hugo
```
vars-common.yml

Turn your app as global:

```
cp -r ~/ondemand/dev/MyCoolApp /var/www/ood/apps/sys/.
```

Set the right permissions!

VIRTUAL DESKTOP



Home / My Interactive Sessions / Desktop

Interactive Apps
Desktops
- Desktop

Servers
- Code Server
- Jupyter Notebook

Desktop

This app will launch an interactive desktop on one or more compute nodes. You will have full access to the resources these nodes provide. This is analogous to an interactive batch job.

Desktop Environment
gnome

Account

Number of hours
8

Number of nodes
1

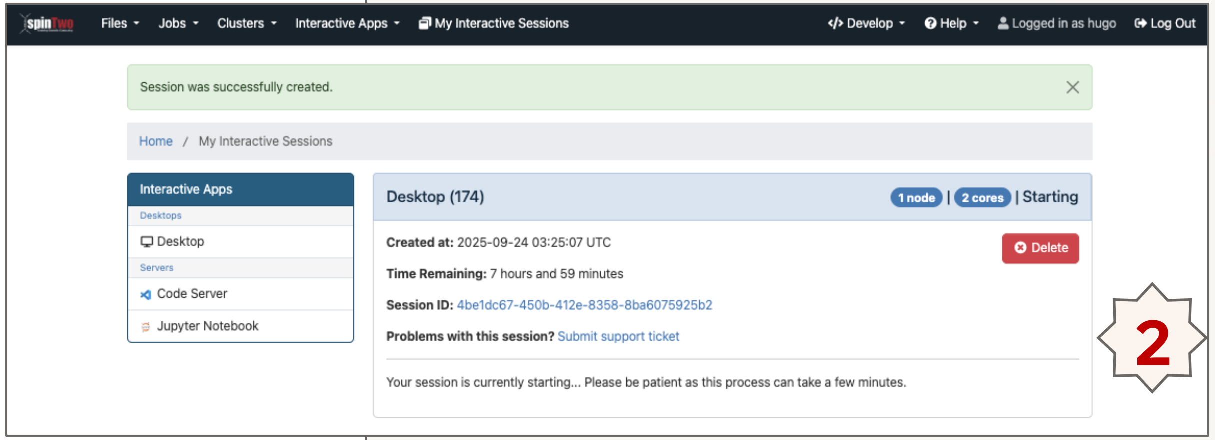
Queue
extended

☑ I would like to receive an email when the session starts

Launch

* The Desktop session data for this session can be accessed under the data root directory.

1



Session was successfully created.

Home / My Interactive Sessions

Interactive Apps
Desktops
- Desktop

Servers
- Code Server
- Jupyter Notebook

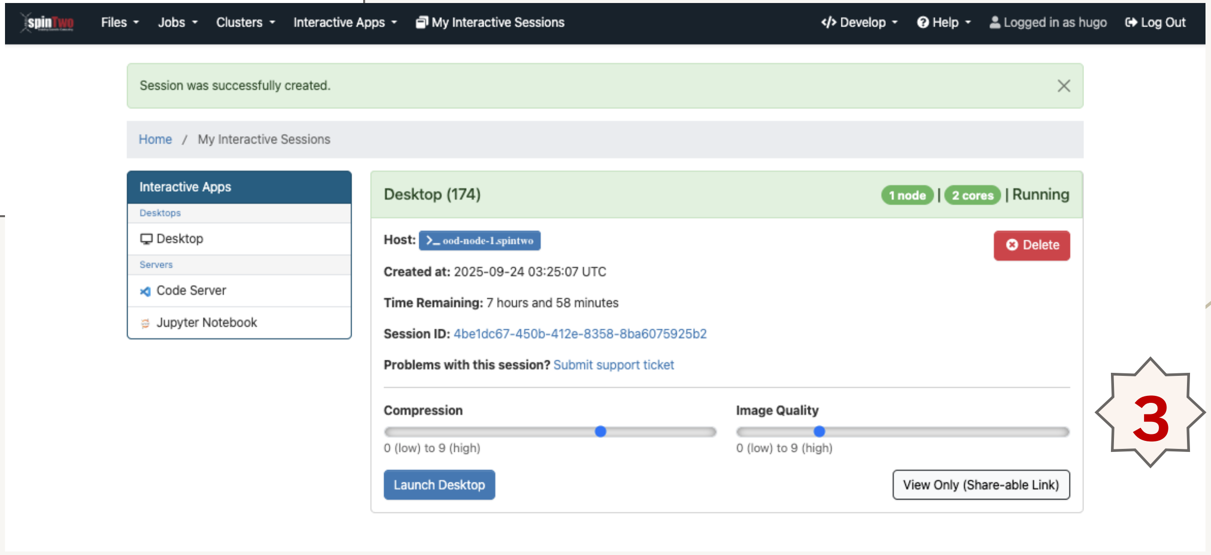
Desktop (174) | 1 node | 2 cores | Starting

Delete

Created at: 2025-09-24 03:25:07 UTC
Time Remaining: 7 hours and 59 minutes
Session ID: 4be1dc67-450b-412e-8358-8ba6075925b2
Problems with this session? Submit support ticket

Your session is currently starting... Please be patient as this process can take a few minutes.

2



Session was successfully created.

Home / My Interactive Sessions

Interactive Apps
Desktops
- Desktop

Servers
- Code Server
- Jupyter Notebook

Desktop (174) | 1 node | 2 cores | Running

Delete

Host: >_ood-node-1.spintwo
Created at: 2025-09-24 03:25:07 UTC
Time Remaining: 7 hours and 58 minutes
Session ID: 4be1dc67-450b-412e-8358-8ba6075925b2
Problems with this session? Submit support ticket

Compression
0 (low) to 9 (high)

Image Quality
0 (low) to 9 (high)

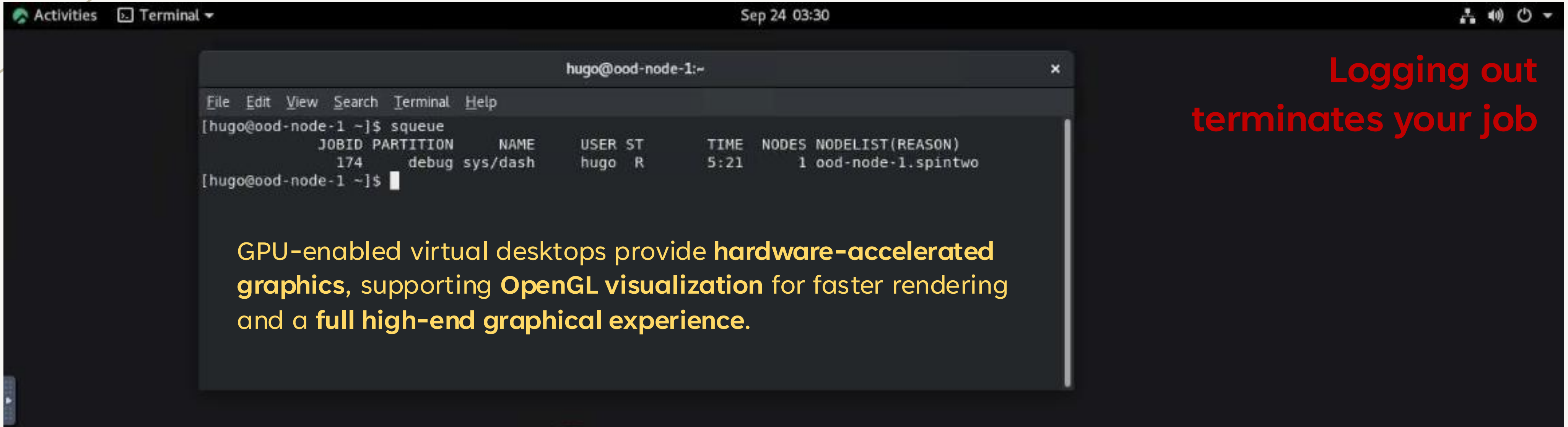
Launch Desktop

View Only (Share-able Link)

3

1. **Request desired resources via Slurm**
2. **Wait for resources to be available**
3. **Launch virtual desktop**

VIRTUAL DESKTOP (CONT.)



```
[hugo@ood-node-1 ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
               174     debug sys/dash     hugo  R       5:21      1 ood-node-1.spintwo
[hugo@ood-node-1 ~]$ 
```

Logging out terminates your job

GPU-enabled virtual desktops provide **hardware-accelerated graphics**, supporting **OpenGL visualization** for faster rendering and a **full high-end graphical experience**.

Key Points

- The **virtual desktop** in Open OnDemand runs on a **dedicated HPC compute node**
- When a user launches a session, OOD allocates a node and starts a **PUN process** that launches the **desktop environment (VNC or ThinLinc)**
- The session uses the node's **CPU or GPU resources**, providing full access to **HPC and visualization** while keeping the **user environment isolated and secure**

Launched sessions can be resumed based on their requested runtime

SETTING UP JUPYTER NOTEBOOK

Application Module Files

jupyter/1.0.lua

```
help([[Jupyter Notebook/Lab]])

whatis("Name: Jupyter")
whatis("Version: 1.0")
whatis("Category: tools")
whatis("Description: Jupyter Notebook and Lab environment")

-- Path to virtualenv or conda environment
local jupyter_root = "/shared/hpc_apps/conda/envs/jupyter-env/"

-- Prepend binary path
prepend_path("PATH", pathJoin(jupyter_root, "bin"))

-- Optional: Add man or lib paths
-- prepend_path("MANPATH", pathJoin(jupyter_root, "share/man"))

-- Optional: Set env vars
setenv("JUPYTER_PATH", pathJoin(jupyter_root, "share/jupyter"))
```

App defined in your HPC system to be run from OOD

OOD Sys application

jupyter/template/script.sh.erb

```
#!/usr/bin/env bash

# Benchmark info
echo "TIMING - Starting main script at: $(date)"

# Set working directory to home directory
cd "${HOME}"

#
# Start Jupyter Notebook Server
#

# Benchmark info
echo "TIMING - Starting jupyter at: $(date)"

module purge
# Launch the Jupyter Notebook Server
set -x

module load jupyter/1.0
jupyter lab --no-browser --config="${CONFIG_FILE}" <%= context.extra_jupyter_args %>
```

```
jupyter/
├── form.yml
├── manifest.yml
├── submit.yml.erb
├── template
│   ├── before.sh.erb
│   └── script.sh.erb
└── completed.{md,html}.erb
```

OOD application template

SETTING UP JUPYTER NOTEBOOK

form.yml

Defines the **job form**, where **dynamic JS attributes adjust options in real-time**—e.g., **enabling GPU choices** only when a GPU partition is selected—**ensuring correct Slurm job submission.**

jupyter/form.yml

```
# Batch Connect app configuration file
#
# @note Used to define the submitted cluster, title, description, and
# hard-coded/user-defined attributes that make up this Batch Connect app.
--- description: |
This app will launch a Jupyter Notebook server on the Dirac slurm cluster.

cluster: "slurm"
form:
  - auto_accounts
  - global_node_type
  - global_partition
  - bc_num_hours
  - cores
  - memory
  - gpus
  - server_type
  - working_dir
  - bc_email_on_started
# Define attribute values that aren't meant to be modified by the user
within
# the Dashboard form
attributes:
  cores:
    widget: "number_field"
    label: "Number of cores"
    value: "1"
    min: 1
    step: 1
```

```
jupyter/
├── form.yml
├── manifest.yml
├── submit.yml.erb
├── template
│    ├── before.sh.erb
│    └── script.sh.erb
└── completed.{md,html}.erb
```

```
memory:
  widget: "number_field"
  label: "Memory per Core (GB)"
  help: "Memory per core requirement"
  min: 1
  max: 8
  step: 1
  required: true
  value: "4"
gpus:
  widget: "number_field"
  label: "Number of GPUs"
  help: "Number of GPUs. Min 1, Max 4"
  min: 1
  max: 4
  step: 1
server_type:
  label: "Jupyter Server type"
  widget: select
    options:
      - ["lab", "lab"]
      - ["notebook", "notebook"]
working_dir:
  widget: "path_selector"
  label: "Working Directory"
  data-target-file-type: dirs # Valid values are: files, dirs, or both
  readonly: false help: "Select your project directory; defaults to $HOME"
```

SETTING UP JUPYTER NOTEBOOK

submit.yml.erb

Translates **user inputs** from the form into **Slurm directives**. It **dynamically generates the sbatch submission script**—e.g., inserting partition, CPUs, memory, or GPU options—so jobs run with the resources selected in the portal.

```
--- title:
”Dirac Desktop App”
cluster:slurm
batch_connect:
  template: vnc
  min_port: 30000
  max_port: 60000

script:
  native:
    - "-N 1”
    - "-n <%= cores.to_i %>”
    - "--mem-per-cpu=<%= memory %>G”
    - "--partition=<%= partition %>_<%= global_node_type %>”
    <%- if bc_email_on_started -%>
    - "--mail-user=<%= ENV['USER'] %>@spintwo.org”
    <%- end %>
```

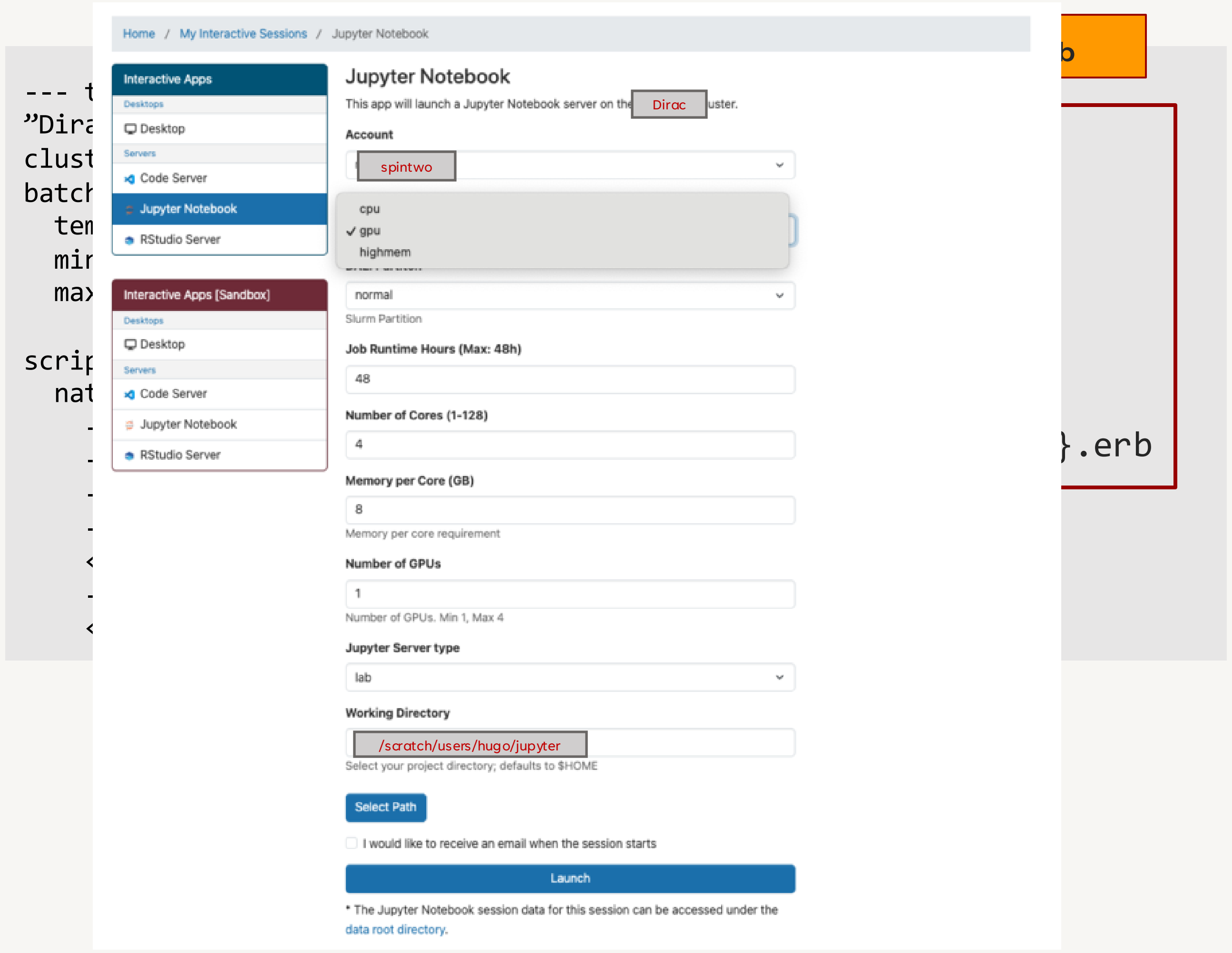
jupyter/submit.yml.erb

```
jupyter/
├── form.yml
├── manifest.yml
├── submit.yml.erb
├── template
│   ├── before.sh.erb
│   └── script.sh.erb
└── completed.{md,html}.erb
```

SETTING UP JUPYTER NOTEBOOK

submit.yml.erb

Translates **user inputs** from the form into **Slurm directives**. It **dynamically generates the sbatch submission script**—e.g., inserting partition, CPUs, memory, or GPU options—so jobs run with the resources selected in the portal.



```
--- t
"Dira
clust
batch
  tem
  mir
  max

scrip
  nat
    -
    -
    -
    -
    <
    -
    <
```

```
b

}.erb
```

Home / My Interactive Sessions / Jupyter Notebook

Interactive Apps
- Desktops
  - Desktop
- Servers
  - Code Server
  - Jupyter Notebook
  - RStudio Server

Interactive Apps [Sandbox]
- Desktops
  - Desktop
- Servers
  - Code Server
  - Jupyter Notebook
  - RStudio Server

Jupyter Notebook

This app will launch a Jupyter Notebook server on the Dirac uster.

Account
spintwo

cpu
✓ gpu
highmem

normal
Slurm Partition

Job Runtime Hours (Max: 48h)
48

Number of Cores (1-128)
4

Memory per Core (GB)
8
Memory per core requirement

Number of GPUs
1
Number of GPUs. Min 1, Max 4

Jupyter Server type
lab

Working Directory
/scratch/users/hugo/jupyter
Select your project directory; defaults to $HOME

Select Path

☐ I would like to receive an email when the session starts

Launch

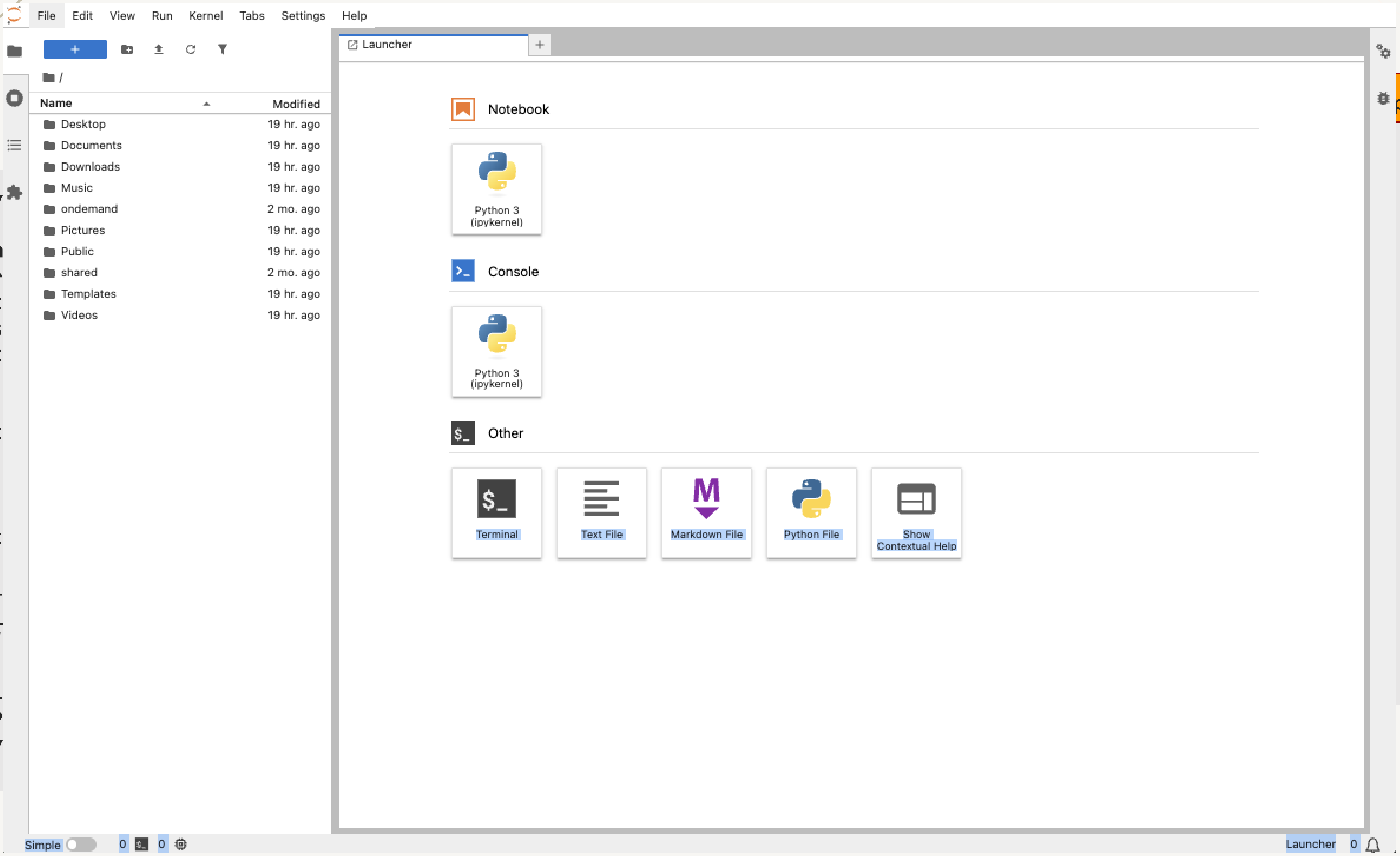
* The Jupyter Notebook session data for this session can be accessed under the data root directory.

SETTING UP JUPYTER NOTEBOOK

```
help([[Jupy

whatis("Nam
whatis("Ver
whatis("Cat
whatis("Des
environment

-- Path to
local jupyt
env/"

-- Prepend
prepend_pat

-- Optional
-- prepend_
"share/man"

-- Optional
setenv("JUP
"share/jupy
```

plate/script.sh.erb

```
extra_jupyter_args %>
```

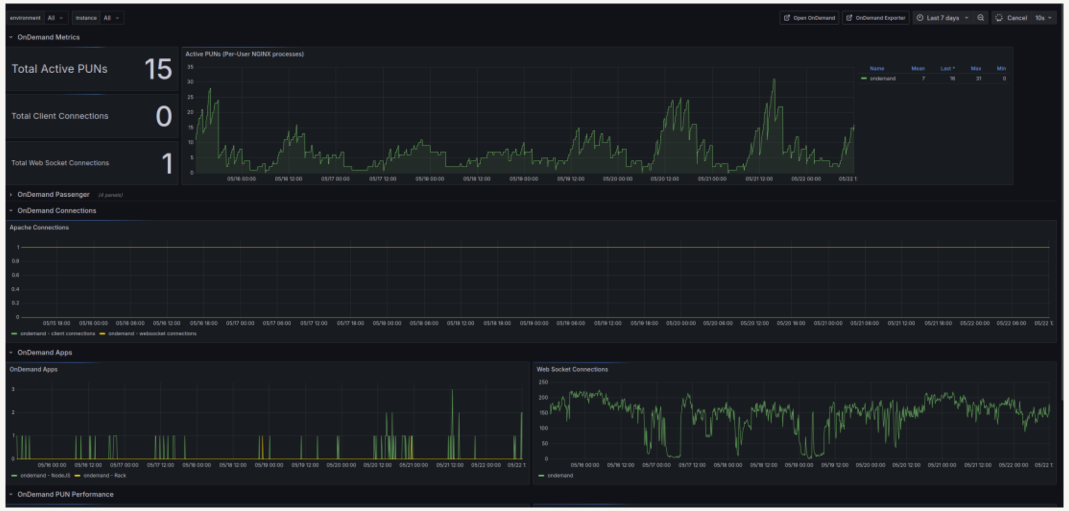


Name	Modified
Desktop	19 hr. ago
Documents	19 hr. ago
Downloads	19 hr. ago
Music	19 hr. ago
ondemand	2 mo. ago
Pictures	19 hr. ago
Public	19 hr. ago
shared	2 mo. ago
Templates	19 hr. ago
Videos	19 hr. ago

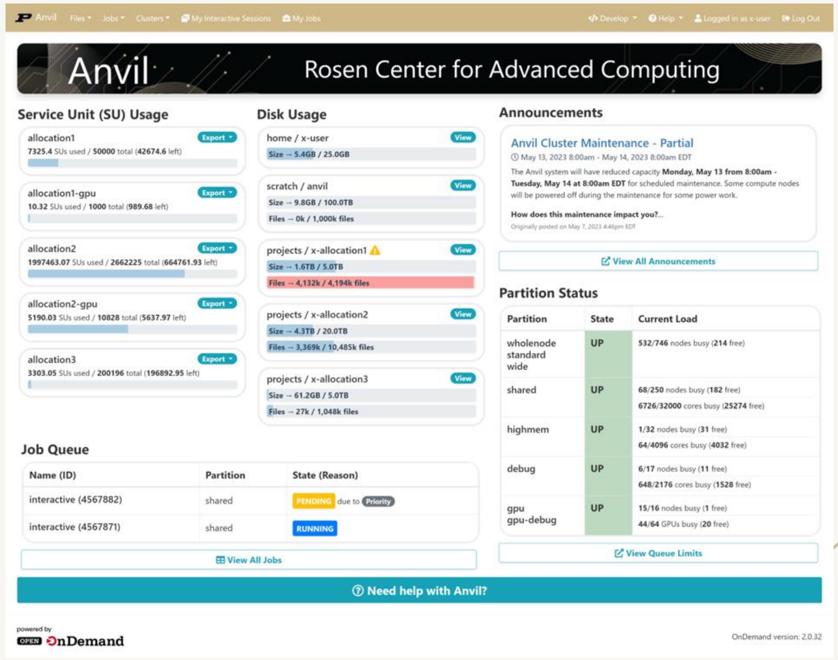
PERFORMANCE AND MONITORING

Open OnDemand provides tools to monitor HPC jobs, interactive sessions, and system performance, with options for tuning, caching, logging, and real-time metrics.

- **Job Monitoring:** View job status, runtime, and resource usage (CPU, memory, GPU) via the Jobs App
- **Interactive Session Monitoring:** Track active Batch Connect apps and virtual desktops
- **Rails & NGINX Tuning:** Optimize Rails thread pool and enable NGINX caching for faster portal response
- **Job & GPU Metrics:** Track job submission latency and GPU/CPU utilization
- **Logging & Integration:** Rails logs, portal logs, centralized logging with ELK/Graylog
- **Cluster Metrics Dashboards:** Prometheus and Grafana visualize node load, memory, disk, and network usage
- **Performance Optimization:** Helps users adjust job parameters or module loads for better efficiency
- **WebSocket Connections:** Monitor active connections to the OOD portal



https://discourse.openondemand.org/t/ondemand-exporter-grafana-dashboard-websocket-metrics/4265



Anvil Rosen Center for Advanced Computing

Service Unit (SU) Usage

allocation1 — 7325.4 SUs used / 50000 total (42674.6 left)

allocation1-gpu — 10.32 SUs used / 1000 total (989.68 left)

allocation2 — 1997463.07 SUs used / 2662225 total (664761.93 left)

allocation2-gpu — 5190.03 SUs used / 10828 total (5637.97 left)

allocation3 — 3303.05 SUs used / 200196 total (196892.95 left)

Disk Usage

home / x-user — Size → 5.4GB / 25.0GB

scratch / anvil — Size → 9.8GB / 100.0TB; Files → 0k / 1,000k files

projects / x-allocation1 — Size → 1.6TB / 5.0TB; Files → 4,132k / 4,194k files

projects / x-allocation2 — Size → 4.3TB / 20.0TB; Files → 3,369k / 10,485k files

projects / x-allocation3 — Size → 61.2GB / 5.0TB; Files → 27k / 1,048k files

Announcements

Anvil Cluster Maintenance - Partial

May 13, 2023 8:00am - May 14, 2023 8:00am EDT

The Anvil system will have reduced capacity **Monday, May 13 from 8:00am - Tuesday, May 14 at 8:00am EDT** for scheduled maintenance. Some compute nodes will be powered off during the maintenance for some power work.

How does this maintenance impact you?...

Originally posted on May 7, 2023 4:46pm EDT

View All Announcements

Partition Status

Partition	State	Current Load
wholenode standard wide	UP	532/746 nodes busy (214 free)
shared	UP	68/250 nodes busy (182 free) 6726/32000 cores busy (25274 free)
highmem	UP	1/32 nodes busy (31 free) 64/4096 cores busy (4032 free)
debug	UP	6/17 nodes busy (11 free) 648/2176 cores busy (1528 free)
gpu gpu-debug	UP	15/16 nodes busy (1 free) 44/64 GPUs busy (20 free)

View Queue Limits

Job Queue

Name (ID)	Partition	State (Reason)
interactive (4567882)	shared	PENDING due to Priority
interactive (4567871)	shared	RUNNING

View All Jobs

Need help with Anvil?

powered by OPEN OnDemand

OnDemand version: 2.0.32

https://cfp.openondemand.org/2025/talk/PBSAAB/

XDMOD INTEGRATION

XDMoD (XD Metrics on Demand) provides **detailed monitoring and reporting for HPC systems.** Integrating it with OOD enables users and admins to **track job efficiency, resource usage, and system performance** from the portal

1) In `/etc/ood/config/nginx_stage.yml`, specify the XDMoD host URL:

```
pun_custom_env:
  OOD_XDMOD_HOST: "https://xdmod.spintwo.org"
```

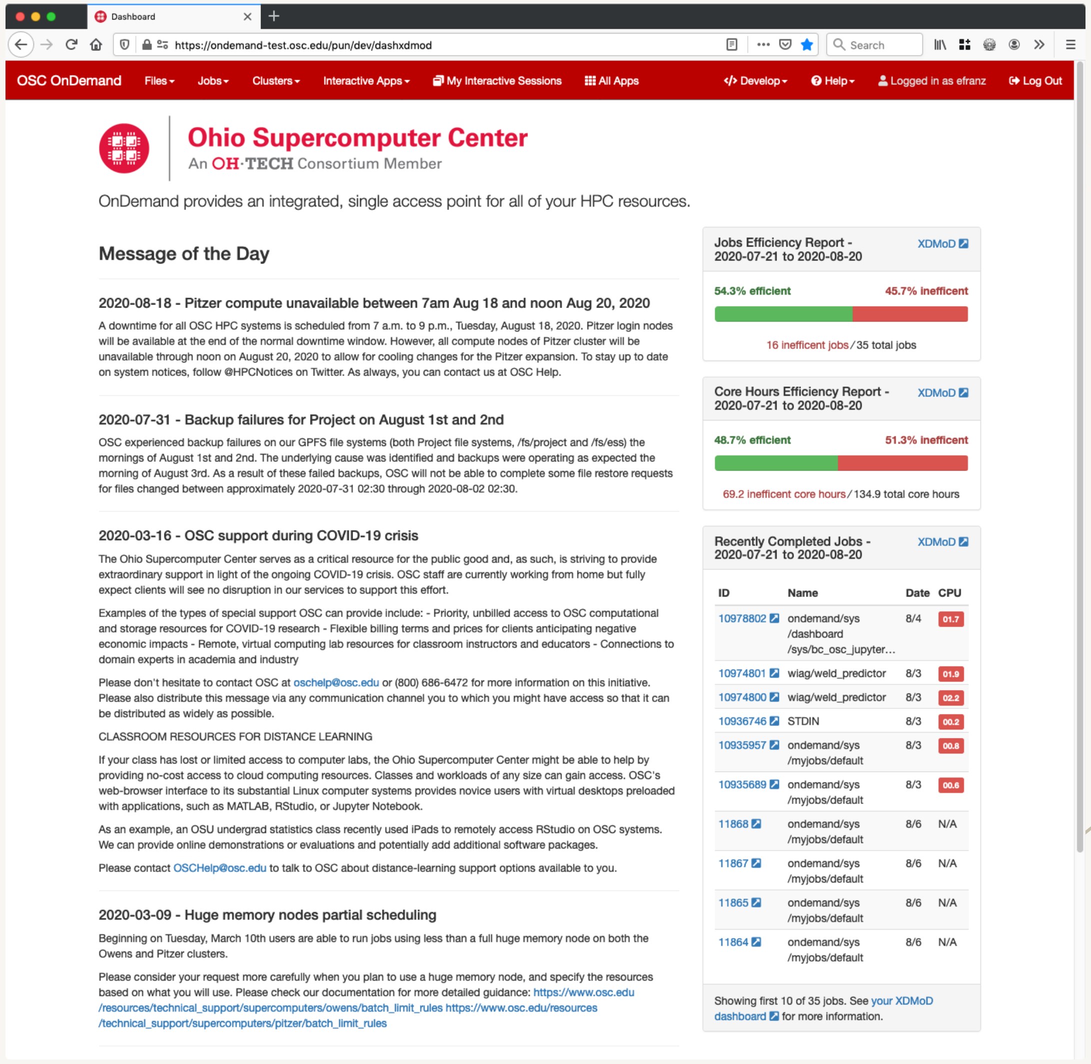
2) In XDMoD's `/etc/xdmod/portal_settings.ini`, configure the CORS domains to permit requests from the OOD portal:

```
[cors] domains = "https://ondemand.spintwo.org"
```

3) Ensure both OOD and XDMoD are configured to use the same Identity Provider (IdP) for SSO. This setup allows users to authenticate once and access both platforms without additional logins.

4) To display XDMoD job metrics in the OOD dashboard, modify the dashboard layout configuration:

```
# /etc/ood/config/ondemand.d/ondemand.yml
dashboard_layout:
 rows:
   - columns:
      - width: 8
        widgets:
          - pinned_apps
          - motd
      - width: 4
        widgets:
          - xdmod_widget_job_efficiency
          - xdmod_widget_jobs
```



OnDemand provides an integrated, single access point for all of your HPC resources.

Message of the Day

2020-08-18 - Pitzer compute unavailable between 7am Aug 18 and noon Aug 20, 2020

A downtime for all OSC HPC systems is scheduled from 7 a.m. to 9 p.m., Tuesday, August 18, 2020. Pitzer login nodes will be available at the end of the normal downtime window. However, all compute nodes of Pitzer cluster will be unavailable through noon on August 20, 2020 to allow for cooling changes for the Pitzer expansion. To stay up to date on system notices, follow @HPCNotices on Twitter. As always, you can contact us at OSC Help.

2020-07-31 - Backup failures for Project on August 1st and 2nd

OSC experienced backup failures on our GPFS file systems (both Project file systems, /fs/project and /fs/ess) the mornings of August 1st and 2nd. The underlying cause was identified and backups were operating as expected the morning of August 3rd. As a result of these failed backups, OSC will not be able to complete some file restore requests for files changed between approximately 2020-07-31 02:30 through 2020-08-02 02:30.

2020-03-16 - OSC support during COVID-19 crisis

The Ohio Supercomputer Center serves as a critical resource for the public good and, as such, is striving to provide extraordinary support in light of the ongoing COVID-19 crisis. OSC staff are currently working from home but fully expect clients will see no disruption in our services to support this effort.

Examples of the types of special support OSC can provide include: - Priority, unbilled access to OSC computational and storage resources for COVID-19 research - Flexible billing terms and prices for clients anticipating negative economic impacts - Remote, virtual computing lab resources for classroom instructors and educators - Connections to domain experts in academia and industry

Please don't hesitate to contact OSC at oschelp@osc.edu or (800) 686-6472 for more information on this initiative. Please also distribute this message via any communication channel you to which you might have access so that it can be distributed as widely as possible.

CLASSROOM RESOURCES FOR DISTANCE LEARNING

If your class has lost or limited access to computer labs, the Ohio Supercomputer Center might be able to help by providing no-cost access to cloud computing resources. Classes and workloads of any size can gain access. OSC's web-browser interface to its substantial Linux computer systems provides novice users with virtual desktops preloaded with applications, such as MATLAB, RStudio, or Jupyter Notebook.

As an example, an OSU undergrad statistics class recently used iPads to remotely access RStudio on OSC systems. We can provide online demonstrations or evaluations and potentially add additional software packages.

Please contact OSCHelp@osc.edu to talk to OSC about distance-learning support options available to you.

2020-03-09 - Huge memory nodes partial scheduling

Beginning on Tuesday, March 10th users are able to run jobs using less than a full huge memory node on both the Owens and Pitzer clusters.

Please consider your request more carefully when you plan to use a huge memory node, and specify the resources based on what you will use. Please check our documentation for more detailed guidance: https://www.osc.edu/resources/technical_support/supercomputers/owens/batch_limit_rules https://www.osc.edu/resources/technical_support/supercomputers/pitzer/batch_limit_rules

Jobs Efficiency Report - 2020-07-21 to 2020-08-20 XDMoD

54.3% efficient 45.7% inefficent

16 inefficent jobs / 35 total jobs

Core Hours Efficiency Report - 2020-07-21 to 2020-08-20 XDMoD

48.7% efficient 51.3% inefficent

69.2 inefficent core hours / 134.9 total core hours

Recently Completed Jobs - 2020-07-21 to 2020-08-20 XDMoD

ID	Name	Date	CPU
10978802	ondemand/sys/dashboard/sys/bc_osc_jupyter…	8/4	01.7
10974801	wiag/weld_predictor	8/3	01.9
10974800	wiag/weld_predictor	8/3	02.2
10936746	STDIN	8/3	00.2
10935957	ondemand/sys/myjobs/default	8/3	00.8
10935689	ondemand/sys/myjobs/default	8/3	00.6
11868	ondemand/sys/myjobs/default	8/6	N/A
11867	ondemand/sys/myjobs/default	8/6	N/A
11865	ondemand/sys/myjobs/default	8/6	N/A
11864	ondemand/sys/myjobs/default	8/6	N/A

Showing first 10 of 35 jobs. See your XDMoD dashboard for more information.

https://osc.github.io/ood-documentation/latest/customizations.html#xdmod-integration

XDMoD INTEGRATION (CONT.)

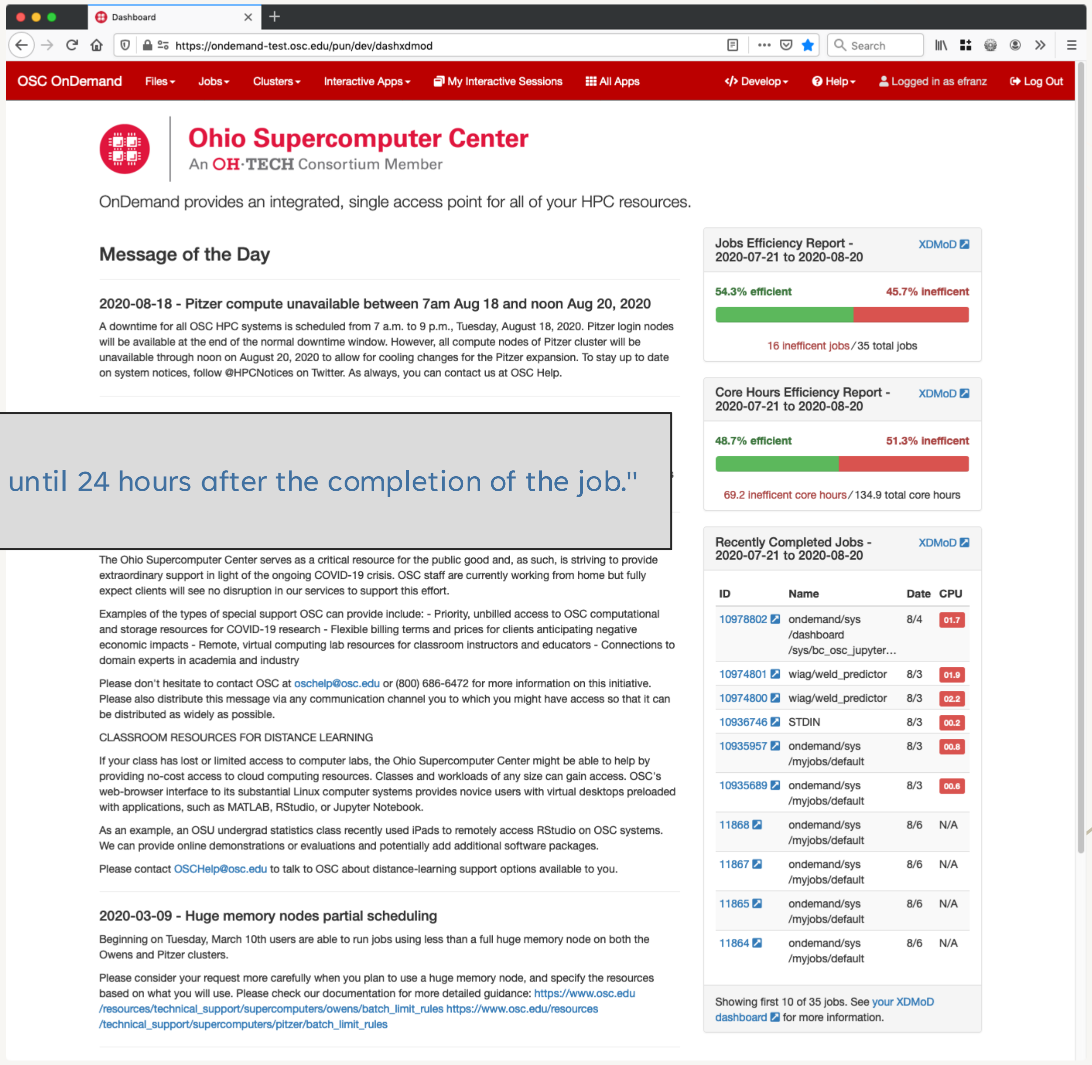
5) In each cluster configuration file add the XDMoD resource ID:

```
custom:
  xdmod:
    resource_id: 1
```

Replace 1 with the actual resource ID assigned to the cluster in XDMoD.

6) In the Job Composer, Open XDMoD job links will include a warning message that the job may not appear in XDMoD for up to 24 hours after the job completed. The message is to address the gap of time between the job appearing as completed in the Job Composer and the job appearing in Open XDMoD after the ingest and aggregation script is run.

```
en:
  jobcomposer: xdmod_url_warning_message: "This job may not appear in Open XDMoD until 24 hours after the completion of the job."
  xdmod_url_warning_message_seconds_after_job_completion: 86400
```



Ohio Supercomputer Center
An OH·TECH Consortium Member

OnDemand provides an integrated, single access point for all of your HPC resources.

Message of the Day

2020-08-18 - Pitzer compute unavailable between 7am Aug 18 and noon Aug 20, 2020

A downtime for all OSC HPC systems is scheduled from 7 a.m. to 9 p.m., Tuesday, August 18, 2020. Pitzer login nodes will be available at the end of the normal downtime window. However, all compute nodes of Pitzer cluster will be unavailable through noon on August 20, 2020 to allow for cooling changes for the Pitzer expansion. To stay up to date on system notices, follow @HPCNotices on Twitter. As always, you can contact us at OSC Help.

The Ohio Supercomputer Center serves as a critical resource for the public good and, as such, is striving to provide extraordinary support in light of the ongoing COVID-19 crisis. OSC staff are currently working from home but fully expect clients will see no disruption in our services to support this effort.

Examples of the types of special support OSC can provide include: - Priority, unbilled access to OSC computational and storage resources for COVID-19 research - Flexible billing terms and prices for clients anticipating negative economic impacts - Remote, virtual computing lab resources for classroom instructors and educators - Connections to domain experts in academia and industry

Please don't hesitate to contact OSC at oschelp@osc.edu or (800) 686-6472 for more information on this initiative. Please also distribute this message via any communication channel you to which you might have access so that it can be distributed as widely as possible.

CLASSROOM RESOURCES FOR DISTANCE LEARNING

If your class has lost or limited access to computer labs, the Ohio Supercomputer Center might be able to help by providing no-cost access to cloud computing resources. Classes and workloads of any size can gain access. OSC's web-browser interface to its substantial Linux computer systems provides novice users with virtual desktops preloaded with applications, such as MATLAB, RStudio, or Jupyter Notebook.

As an example, an OSU undergrad statistics class recently used iPads to remotely access RStudio on OSC systems. We can provide online demonstrations or evaluations and potentially add additional software packages.

Please contact OSCHelp@osc.edu to talk to OSC about distance-learning support options available to you.

2020-03-09 - Huge memory nodes partial scheduling

Beginning on Tuesday, March 10th users are able to run jobs using less than a full huge memory node on both the Owens and Pitzer clusters.

Please consider your request more carefully when you plan to use a huge memory node, and specify the resources based on what you will use. Please check our documentation for more detailed guidance: https://www.osc.edu/resources/technical_support/supercomputers/owens/batch_limit_rules https://www.osc.edu/resources/technical_support/supercomputers/pitzer/batch_limit_rules

Jobs Efficiency Report - 2020-07-21 to 2020-08-20 XDMoD

54.3% efficient 45.7% inefficent

16 inefficent jobs / 35 total jobs

Core Hours Efficiency Report - 2020-07-21 to 2020-08-20 XDMoD

48.7% efficient 51.3% inefficent

69.2 inefficent core hours / 134.9 total core hours

Recently Completed Jobs - 2020-07-21 to 2020-08-20 XDMoD

ID	Name	Date	CPU
10978802	ondemand/sys /dashboard /sys/bc_osc_jupyter...	8/4	01.7
10974801	wiag/weld_predictor	8/3	01.9
10974800	wiag/weld_predictor	8/3	02.2
10936746	STDIN	8/3	00.2
10935957	ondemand/sys /myjobs/default	8/3	00.8
10935689	ondemand/sys /myjobs/default	8/3	00.6
11868	ondemand/sys /myjobs/default	8/6	N/A
11867	ondemand/sys /myjobs/default	8/6	N/A
11865	ondemand/sys /myjobs/default	8/6	N/A
11864	ondemand/sys /myjobs/default	8/6	N/A

Showing first 10 of 35 jobs. See your XDMoD dashboard for more information.

https://osc.github.io/ood-documentation/latest/customizations.html#xdmod-integration

SECURITY CONSIDERATIONS

Ensuring **secure access to HPC resources** requires encryption, controlled user permissions, and careful management of sessions and data.

⚠ **Backend Service Communication:** Client-to-server traffic uses TLS, but connections to backend services (e.g., Jupyter, VS Code) currently use unencrypted HTTP. The OOD team is working to secure these internal connections.

⚠ **Reverse Proxy Injection Risk:** Enabling reverse proxying can allow malicious URL injection. This risk is mitigated by setting a strict `host_regex` in Apache to allow only defined backend nodes, ideally including their domain for maximum security.

Secure Proxy: Only allow nodes with the host_regex regex pattern to servers via proxy (in `/etc/ood/config/ood_portal.yml`):

```
node_uri: "/node"
rnode_uri: "/rnode"
host_regex: node\d+\.spintwo\.com'
```

⚠ **Interactive Application Credentials:** Each OOD app can generate random passwords for sessions, stored in plain text in the user's job directory. Because these are human-readable, they must be treated as sensitive. Proper permissions and secure filesystem configuration are critical to prevent unauthorized access to running applications and their data.

⚠ **Restrict SSH key authentication** whenever possible by controlling usage, minimizing risk, and protecting cluster security.

Open OnDemand is built with security as a foundational principle, supporting robust authentication and minimizing risks through per-user NGINX (PUN) sessions that run as the authenticated user rather than root.

BEST PRACTICES

The following are considered fundamental actions or configurations that should be followed to ensure a successful and secure deployment of OOD:

- Verifying hardware and software compatibility ensures a stable environment for OOD to function correctly. This is crucial to avoid compatibility issues.
- Using **HTTPS**, secure authentication methods (**SSO**, LDAP, MFA), and logging practices are foundational to protecting users and the system from unauthorized access and breaches.
- Keep your `oidc_client_id` and `oidc_client_secret` secured.
- **Version control your configuration. Use Git to track changes to `yml` files and app configurations. Automate with Ansible and CI/CD pipelines.**
- Use the latest stable version of Open OnDemand available (via the OOD YUM repository).
- **Ensure the file permissions inside `~/ondemand` and other accessible per user directories are set correctly to only allow the intended user(s) and groups with access.** File permissions can disallow unintended access from applications beyond Open OnDemand.
- **Restrict SSH access to only authorized users and required HPC hosts**; encourage to use SSH authorization keys instead of passwords. Restricting SSH access is crucial for maintaining the security of your HPC, especially when users are accessing systems through a web interface.
- Implementing monitoring and logging is critical for maintaining system performance, diagnosing issues, and auditing activities. **Regular security and compliance audits**.
- **Use MOTD & portal announcements for user guidance.**
- Use `/etc/ood/config/ondemand.d/global_bc_items.yml` to define common interactive application form items. For example, it is likely that all applications will share the same partition definitions. To avoid copy/paste of the same form values, the partitions can be defined globally as

```
global_bc_form_items:
  global_partitions:
    widget:"select"
    options:
      -[
          "normal",
          "normal",
          data-max-bc-num-hours-for-cluster-dirac:240,
          data-max-bc-num-hours-for-cluster-feynman:8
      ]
```

  then, in the interactive app form:

```
form:
  -global_partitions
```

- **Interactive application form options should be limited to what makes sense for the given application.** For example, an application like VSCode does not need to run on a long-running partition or require a massive amount of RAM for it to function properly.
- Create custom templates for common job types (e.g., MPI, GPU, serial) and store the templates in `/etc/ood/config/apps/myjobs/templates`.

SUMMARY

Open OnDemand provides secure, browser-based access to HPC resources, enabling interactive applications, file management, and automated, maintainable deployments.

- ✓ **Open OnDemand as an Integrator**: Provides a unified portal for all HPC tools and resources available in your organization.
- ✓ **Browser-Based HPC Access**: Open OnDemand supports CLI and GUI users with full SSO, interactive apps, file management, and messaging.
- ✓ **Interactive & Batch Applications**: Launch CLI, GUI, and containerized apps with Slurm integration.
- ✓ **Automation & Maintainability**: Ansible and containerized apps simplify deployment and maintenance of HPC environments.
- ✓ **Secure, Optimized Integration**: Ensures secure HPC access, workflow support, and adherence to best practices for scalable, user-friendly systems.

THANK YOU

Hugo Hernández – hugo@spintwo.com

https://spinTwo.com

spin Two

Enabling Scientific Computing