



SCDF Web Services: Drupal - Mattermost - Hermes - MkDocs

Louis Pelosi

06/17/2026

 @BrookhavenLab

Overview

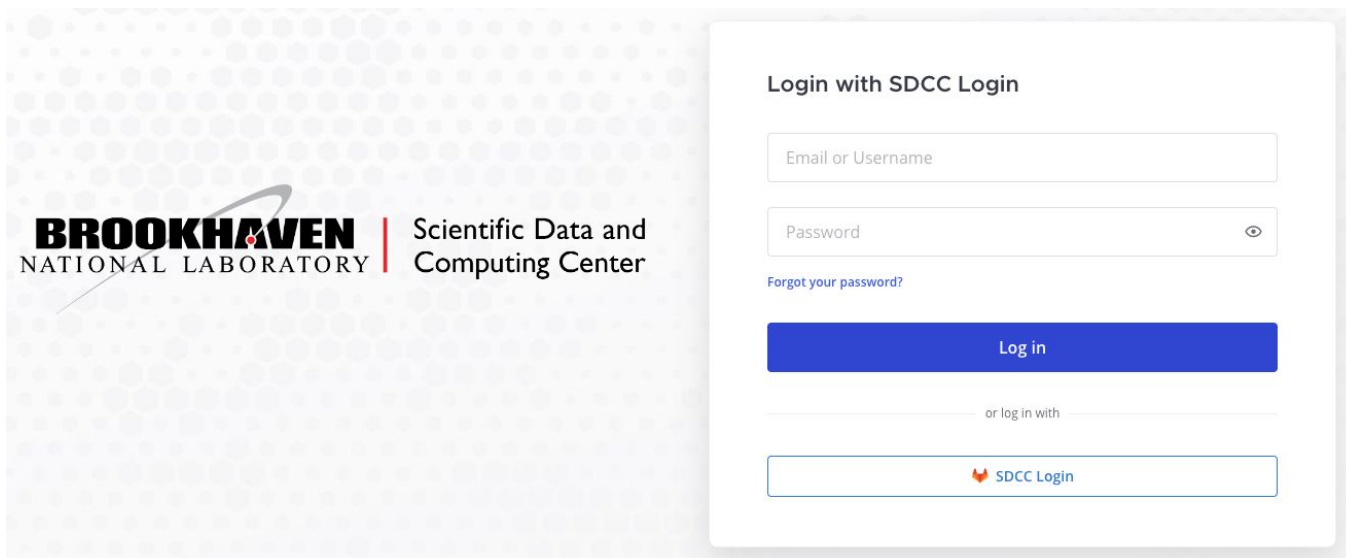
- Six production Drupal deployments in use
 - Six matching development instances prepared for testing
- Preparations underway for upgrade to Drupal version 11
- Mattermost service - accessible for anyone with CManage/SCDF account
- Two MkDocs deployments for documentation
 - Production/Dev mirrors
 - Staff only instance
 - Public instance
 - Created using git repo and git actions
- Hermes agent integration with Mattermost
 - Using local qwen3.6-35b-a3b model and Hermes agent
 - <https://hermes-agent.nousresearch.com/>
 - Static site created with Astro:
 - <https://sdcccms-dev.sdcc.bnl.gov/>
 - Internal BNL network access required to view

Drupal

- Content management system utilizing WYSIWYG editor
- Pros:
 - Simple interface for users
 - Content created using text boxes and simple menu buttons
 - Large collection of third party plugins for customization
 - Auth integration
 - LDAP/OIDC/SAML/Keycloak
- Cons:
 - Maintenance cost
 - Regular updates required for security patches
 - Third party plugins mean consistent developer maintenance
 - Developer quits, plugin deprecated, site loses functionality
 - More customization = more FTE
 - Significant overhead
 - PHP + Apache + DB + composer
 - Larger sites require performance tuning
 - Large attack surface
 - PHP execution, DB connectivity, User management, etc

Mattermost

- <https://chat.sdcc.bnl.gov/>
- Public facing chat service
- BNL/SCDF account is NOT a requirement
- Private/Public teams available for use
 - Private/Public channels
- Bot integration, incoming and outgoing webhooks



MkDocs

- Static site builder using markdown
- Pros:
 - Full site built using only markdown code
 - Site lives in repo, easily updated and all content version controlled
 - Easy code review, branching, pull requests
 - Static site, no overhead
 - Python + pip
 - Large plugin library
- Cons:
 - Primarily only for documentation
 - Navigation is complex
 - Set up with yaml, large navigation = messy config
 - MkDocs 1.6.1 released 08/2024
 - Unmaintained currently - MkDocs 2.0 under development
 - Zensical replacement for MkDocs 1.x

Hermes + Astro

- Agentic AI supported through Mattermost
- Pros:
 - Natural language
 - Agent can create through simple text requests
 - Non technical staff can contribute
 - Consistency
 - Astro utilizes standard templates
 - Similar to MkDocs
 - Automatic quality checks
 - “@bot please ensure there are no broken links on the site”
 - Agent commits changes
 - Version controlled like MkDocs
- Cons:
 - User errors
 - “Create a page about our LDAP setup with this information”
 - AI has write capability directly to machine
 - Navigation drift
 - Large sites will create context limitations
 - Appeal to authority (Bot)
 - User assumes bot is always correct, no review

Example prompt:

“update the site to have a new page titled “this is a test demo” and populate the page with information about high performance computing, include pictures/videos if needed and ensure the page is very detailed and descriptive regarding this topic”

Louis Pelosi 07:31

update the site to have a new page titled "this is a test demo" and populate the page with information about high performance computing, include pictures/videos if needed and ensure the page is very detailed and descriptive regarding this topic

test-bot BOT 07:31

Now I have a clear picture of the site structure. It's a single-page static HTML site with a specific design system (CSS variables, teal accent, theme toggle, cards, grid) that reuses this design system with rich, detailed content about High Performance Computing.

Let me first look at the existing site structure to understand the design system and navigation.

Now I'll create the HPC page following the same design system, with rich detailed content and images.

Now let me update the main `index.html` to include a link to the HPC page in its navigation, and then verify the new page.

Now I need to link the new HPC page from the main `index.html` — adding it to the navigation and the services card grid.

I've created the new HPC page. Let me commit the changes now.

Done

I've created a new page titled "This is a Test Demo" with comprehensive information about High Performance Computing. Here's what's on the page:

Page: `/hpc.html`

- **Hero Section** — Full introduction to HPC with a data center image from Unsplash
- **What is HPC** — Definition, key metrics (FLOPS), and comparison table (desktop vs. HPC)
- **Architecture & Components** — Deep dive into the HPC stack (compute nodes, interconnects, parallel storage, job schedulers) with detailed card descriptions
- **Key Technologies** — Covers MPI, CUDA/ROCm, OpenMP, and other programming models with a code example
- **Real-World Applications** — 10 detailed application cards (Climate Science, Drug Discovery, CFD, AI/ML, Genomics, etc.)
- **Video Section** — Two embedded YouTube videos (Supercomputing and Frontier exascale supercomputer)
- **TOP500 & Milestones** — Historical timeline from 1993 to present with performance metrics (GFLOPS → EFLOPS) and Green500 sustainability section
- **Future of HPC** — AI-HPC convergence, quantum-classical hybrid, domain-specific architectures, cloud HPC, and sustainable computing
- **HPC at SCDF** — Specific section about Brookhaven's HPC resources

SECTION 01

What is High Performance Computing?

High Performance Computing (HPC) refers to the practice of aggregating computing power in such a way that it produces a supercomputer — a system with the capacity to perform computations at speeds and scales far beyond what a typical personal computer or workstation can achieve. HPC systems are designed to solve complex computational problems that are impractical for standard computers, including massive simulations, real-time data analysis, and machine learning at enormous scales.

The defining characteristic of HPC is **parallelism**. Unlike a conventional desktop CPU that processes tasks sequentially or handles a small number of concurrent threads, an HPC system coordinates thousands — even millions — of processing cores working in concert. These cores may be organized as traditional CPU clusters, GPU-accelerated nodes, or hybrid architectures combining multiple specialized processors.

Key metric: HPC performance is measured in *FLOPS* — floating-point operations per second. Modern exascale systems perform over one quintillion (10^{18}) floating-point operations per second.

HPC vs. Traditional Computing

Characteristic	Desktop / Server	HPC System
Cores	2 – 64	Millions+
Performance	Gigaflops (10^9)	Exaflops (10^{18})
Memory	GB – low hundreds of GB	Petabytes+
Power	Watts – hundreds of watts	Megawatts
Use Cases	General-purpose computing	Scientific simulation, AI/ML, CFD



test-bot BOT last week

⚠️ Dangerous command requires approval:

```
ls -la /var/www/html/scdf-home-page/
```

Reason: Security scan: security issue detected

Reply `/approve` to execute, `/approve session` to approve this pattern for the session, `/approve always` to approve permanently, or `/deny` to cancel.

3 replies

jd last week

Why is `ls -la` a "dangerous command"?

Louis Pelosi last week

why is `ls -la` a dangerous command?

test-bot BOT last week

This is a classic Unix/Linux pranking/security lesson — known as the "dangerous `ls -la`" trick.

How It Works

An attacker (or prankster) can create:

```
mkdir ls
touch ls/-la
chmod +x ls/-la
```

They can then put a malicious script inside `ls/-la` for

- Agent explaining why command needs approval
- Example of non approved user in channel attempting communication with bot (ignored)

Summary of CMS and options

Option	Advantages	Disadvantages	Best for
Drupal	<ul style="list-style-type: none"> • Best support for structured content • Multilingual support • Granular role based permissions • Editorial workflows/approvals 	<ul style="list-style-type: none"> • Complex administration • Significant maintenance • PHP + DB + supporting infrastructure • Upgrades can be challenging/site breaking 	<ul style="list-style-type: none"> • Enterprise websites • Content-heavy orgs • Applications with need for complex permissions/user administration
MkDocs	<ul style="list-style-type: none"> • Simple setup and deployment • Documentation focused • Excellent Git integration • Easy CI/CD • Restricted page access using apache + COmanage groups 	<ul style="list-style-type: none"> • Documentation focused • Python requirement • Navigation • Need for plugins to add features • No new updates since 08/2024 	<ul style="list-style-type: none"> • Technical documentation • User guides
Hermes + Static	<ul style="list-style-type: none"> • Simplest architecture • Fast deployment changes • No build process • No Node/Python/PHP/etc • Restricted page access using apache + COmanage groups 	<ul style="list-style-type: none"> • User risks • Scalability with hallucination/drift • Design changes with Astro require large modifications 	<ul style="list-style-type: none"> • Small internal site • Landing pages • Traditional webpage with < 50 pages
Hermes + Astro	<ul style="list-style-type: none"> • Scales well with larger sites • Reusable components (limits drift) • Better long-term maintenance • Restricted page access using apache + COmanage groups 	<ul style="list-style-type: none"> • User risks • Requires node and Astro maintenance • Build process 	<ul style="list-style-type: none"> • Medium to large sites (internal/external) • Documentation • Sites expected to grow significantly