AppHub User Manual

An Introductory Guide to the AppHub Science Gateway

https://apphub.wolke.uni-greifswald.de

1. Introduction & Home

AppHub is the Science Gateway of the University of Greifswald, providing computational resources to all users. It offers access to high-performance systems for data analysis, simulation, and other computational tasks — available to users from all fields including physics, chemistry, biology, and beyond.

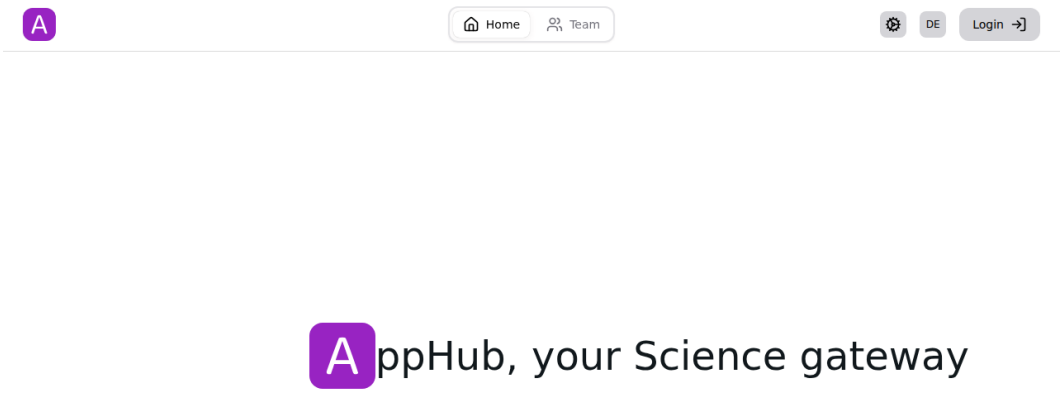


Figure: AppHub Home Page — the main landing page and science gateway

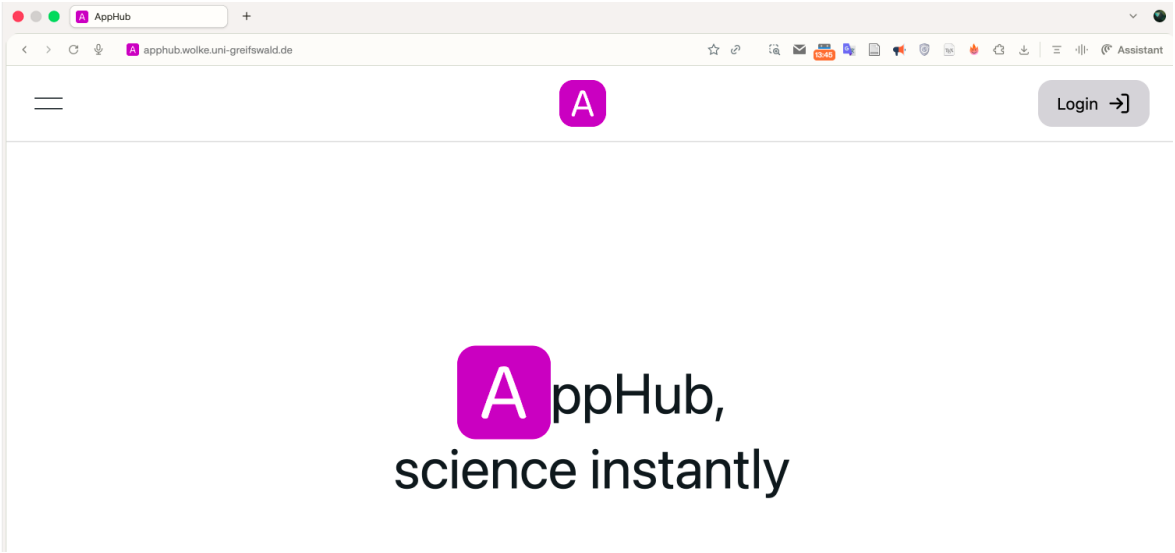
Key Features at a Glance

- Software: Datascience, QGIS, Blender, Statistics, and many more domains
- Interfaces: VS Code, JupyterLab, RStudio, Desktop
- Hardware: 50 Servers, 1520 vCPUs, 8 TB RAM, 60 GPUs, 100 GB per user
- AI Services: Text, Image, Audio, API and more
- Resources available for all the universities in Mecklenburg Vorpommern and guest account holders.

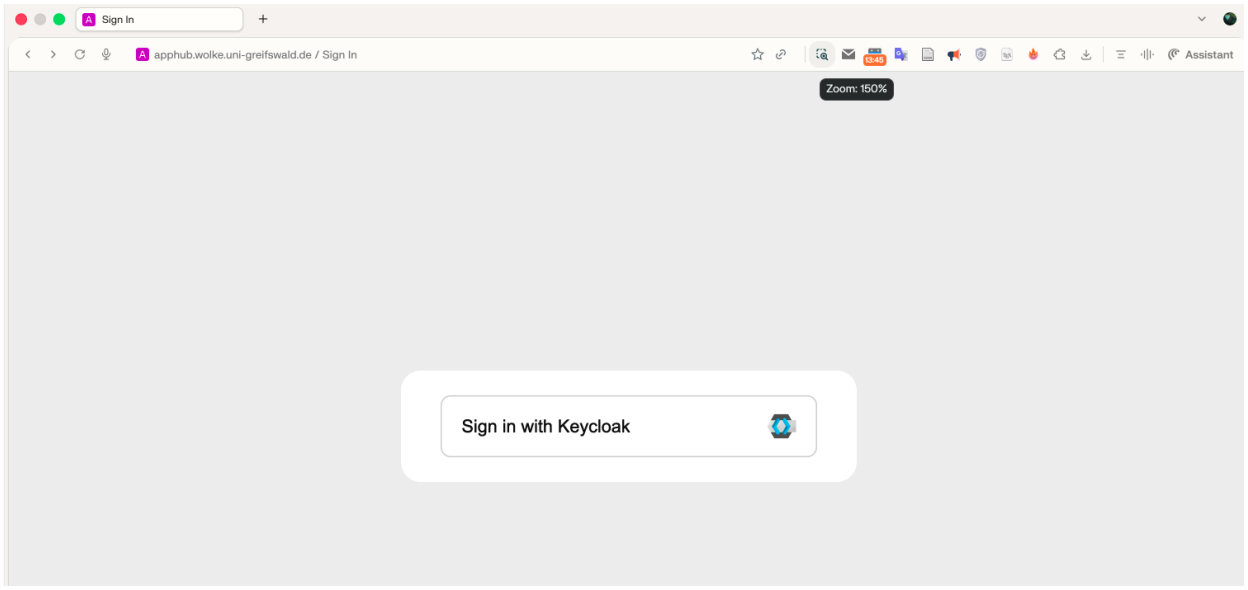
How It Works (Overall)

AppHub is designed to be user-friendly, scalable, and flexible. It supports popular programming languages such as Python and R, handles large datasets, and ensures reproducibility by recreating every server from exactly the same software revision.

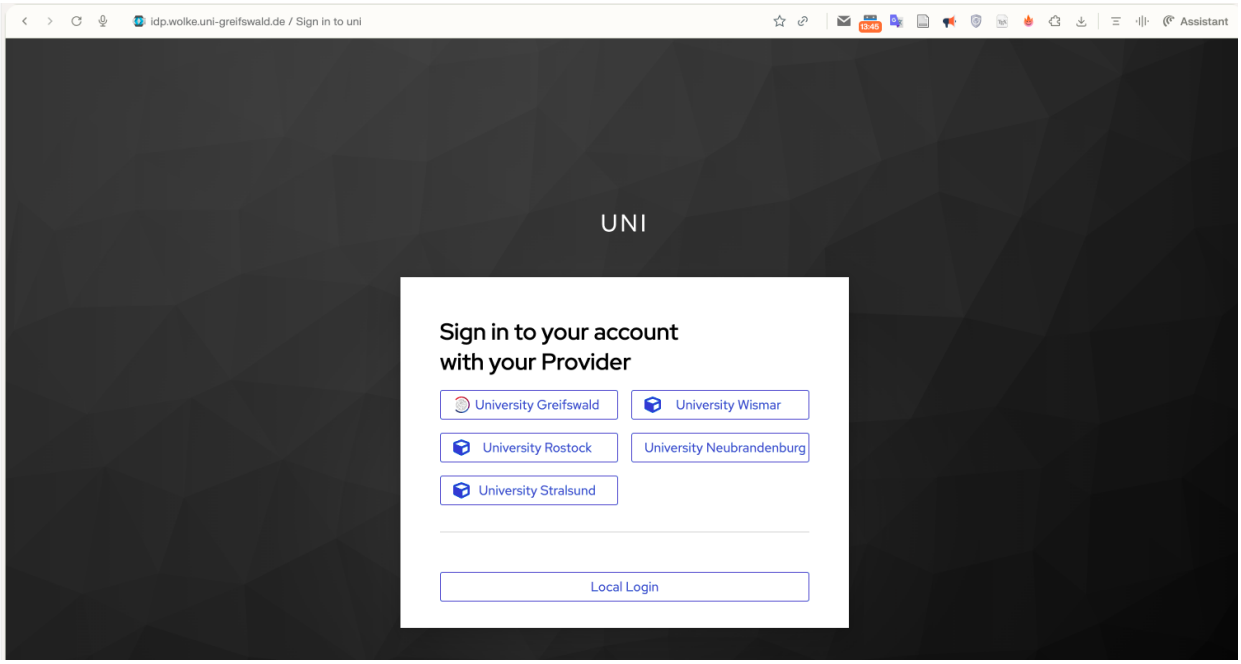
- Step 1: Navigate to https://apphub.wolke.uni-greifswald.de and press Login in the top right corner.



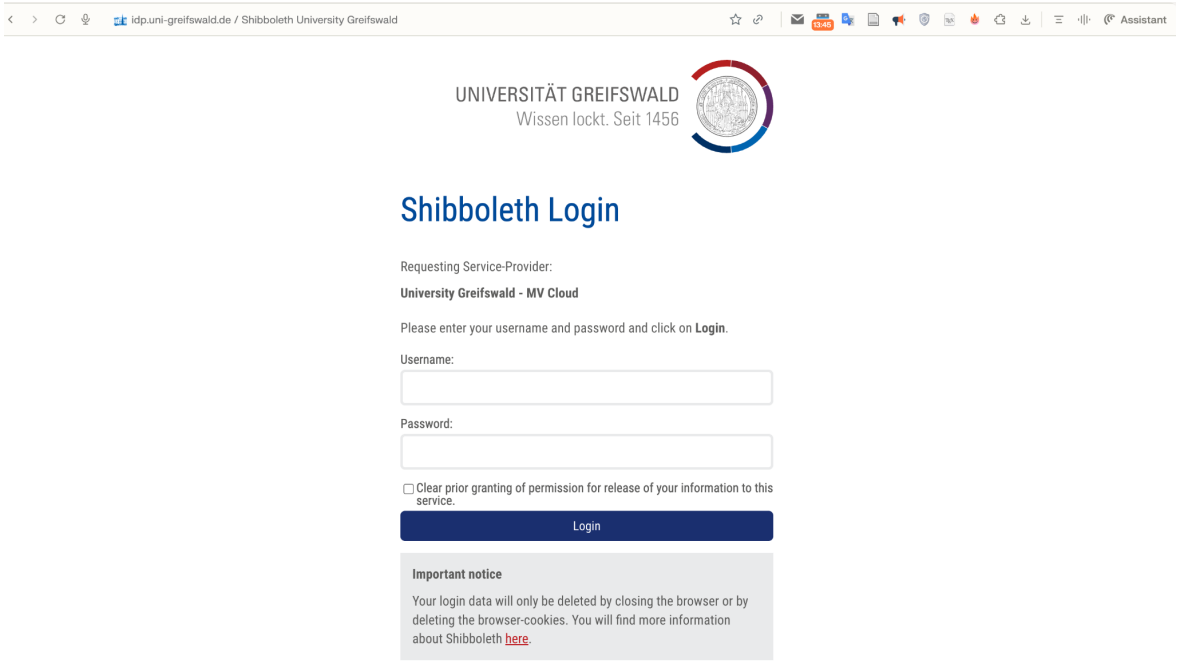
- Step 2: Press on Sign in with Keycloak



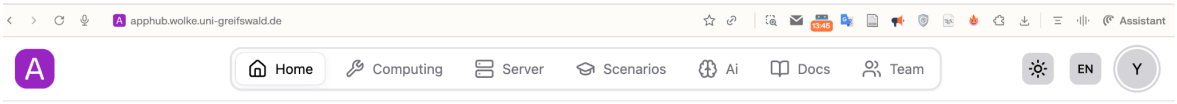
- Step 4: Select your University (or Guest account University)



- Step 5: Log in with your university credentials

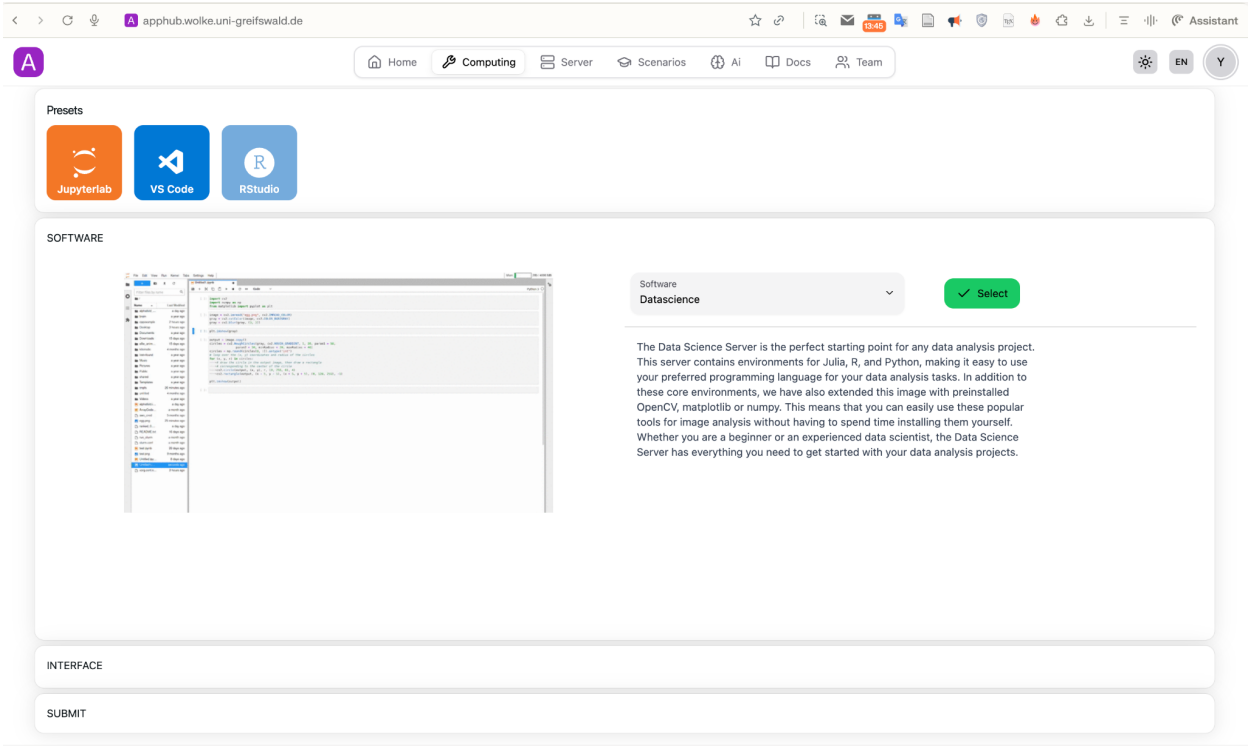


- Step 3: Select Computing to launch a computing environment

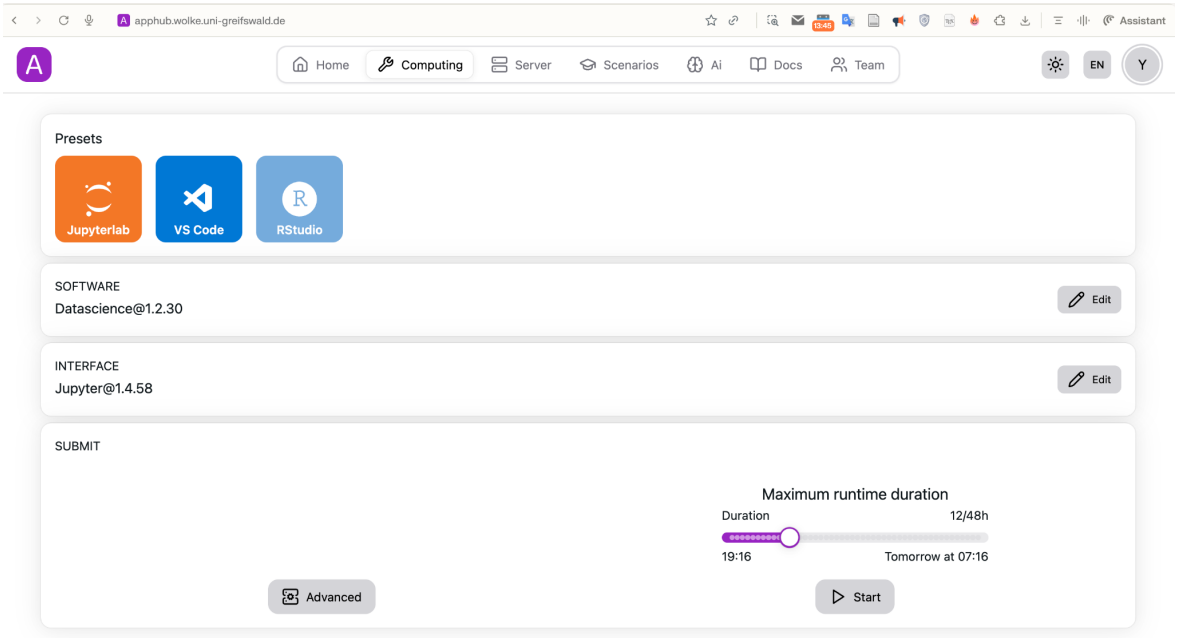


AppHub, your Science gateway

- Step 4: Choose your Software, Interface, and configuration



- Step 5: Click Start to launch your server



2. Computing

The Computing section is the core of AppHub. It guides you through a step-by-step wizard to configure and launch your personal computing server. Navigate to the Computing tab in the top navigation bar to begin.

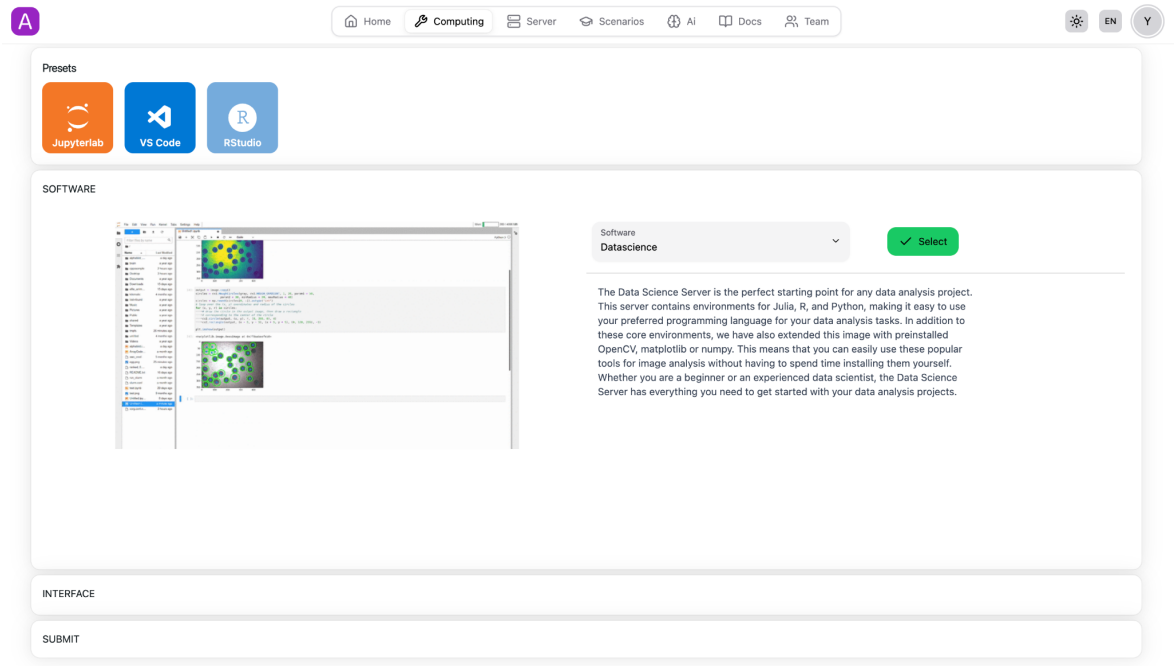


Figure: Computing Page — overview showing Presets, Software, Interface, and Submit sections

2.1 Presets

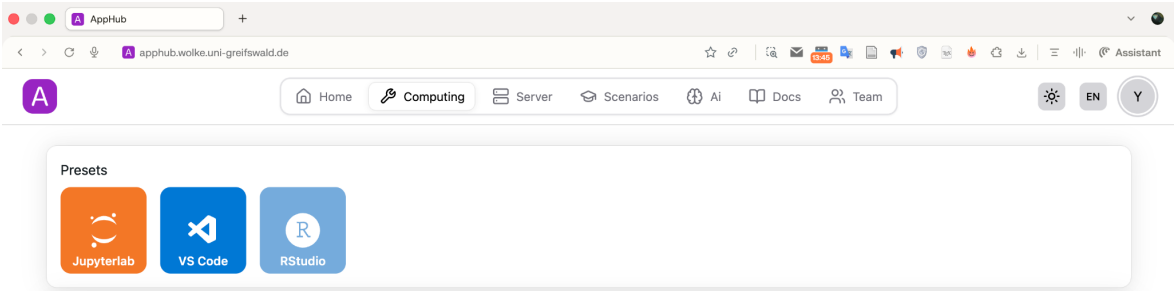


Figure: Computing — Preset selection options: JupyterLab, VS Code, and RStudio

Presets allow you to quickly set up a computing environment with one click. Three preset configurations are available:

- JupyterLab — A web-based interactive notebook environment ideal for data science, Python/R/Julia programming, and exploratory analysis.

- VS Code — A full-featured code editor environment, perfect for software development, scripting, and debugging in any language.
- RStudio — A dedicated environment for R programming, statistical analysis, and data visualization.

Tip: Selecting a preset automatically sets both the Software and Interface fields. You can still customize them in the next steps.

2.2 Software Selection

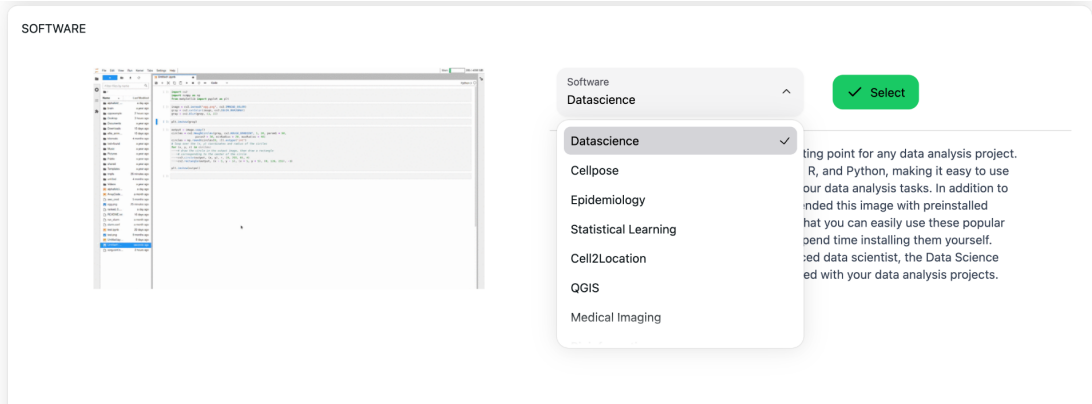


Figure: Software Selection options: Datascience, Cellpose, Epidemiology, etc.

The SOFTWARE section lets you choose the domain or software stack for your computing environment. Click on the SOFTWARE accordion to expand it, then use the dropdown to browse and select your desired software.

- Datascience — Python, R, Julia packages for machine learning and data analysis
- Cellpose — Cell segmentation and microscopy image analysis
- Epidemiology — Tools for examining health data
- Statistical Learning — Packages for statistical modeling and machine learning
- Cell2Location — Spatial transcriptomics tools
- QGIS — Geographic Information System for spatial data analysis
- Medical Imaging — Tools for medical image processing
- Bioinformatics — Sequence analysis and genomics pipelines
- Biochemistry — Molecular modeling and biochemical analysis
- Computational Chemistry — Biomolecular simulation and cheminformatics.
- Octave — MATLAB-compatible numerical computing
- Graphics — Visualization and rendering tools
- Genetics, Matlab, Pharmacy, Metashape, QtCreator, ROS 2, Statistics, IDL, Tensorflow, Deep Learning

After selecting software, click the Select button to proceed to the Interface section.

2.3 Interface Selection

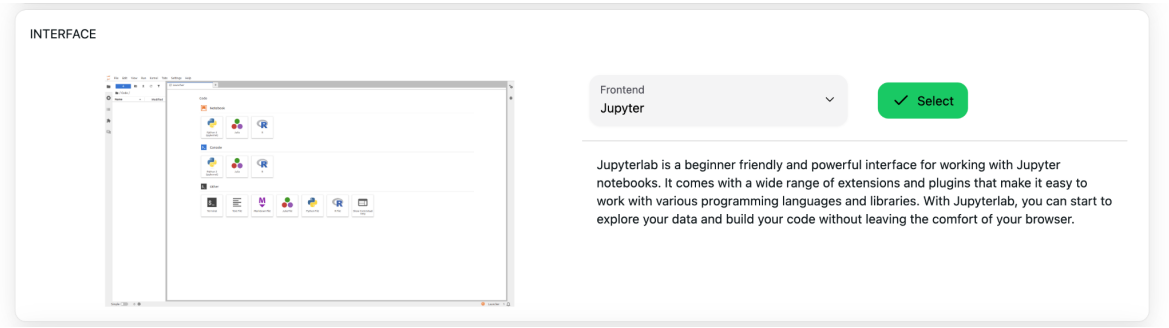


Figure: Interface Selection options: Code, JupyterLab, RStudio and Desktop

Once you have selected your software, the INTERFACE section becomes active. Here you choose the frontend (graphical interface) you want to use to interact with your computing session.

- Code (VS Code) — Visual Studio Code in the browser; supports all languages, extensions, and debugging tools. Best for general development work.
- Jupyter (JupyterLab) — Interactive notebook environment; supports Python, R, Julia; ideal for data analysis, visualization, and sharing results.
- RStudio — Integrated development environment for R; includes console, script editor, plots, and package manager.
- Desktop — Full-featured Ubuntu desktop environment.

Each interface shows a description and preview when selected. After choosing, click the Select button to advance to the Submit section.

2.4 Submit Configuration

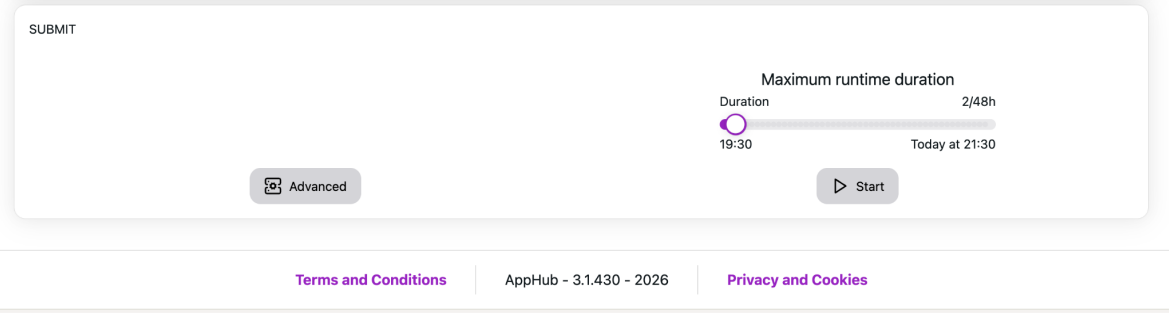


Figure: Submit Configuration

The SUBMIT section is where you finalize the runtime settings before launching your server.

Maximum Runtime Duration

Use the slider to set how long your server will run before automatically stopping. The range is 2 to 48 hours.

- Minimum: 1 hours — for quick, short tasks

- Default: Typically set to a moderate value
- Maximum: 48 hours — for long-running computations, training jobs, or overnight processing

Note: When the runtime expires, your server is stopped. Make sure to save your work before the timer runs out, or use the Prolong function from the Server tab to extend it.

2.5 Advanced Settings

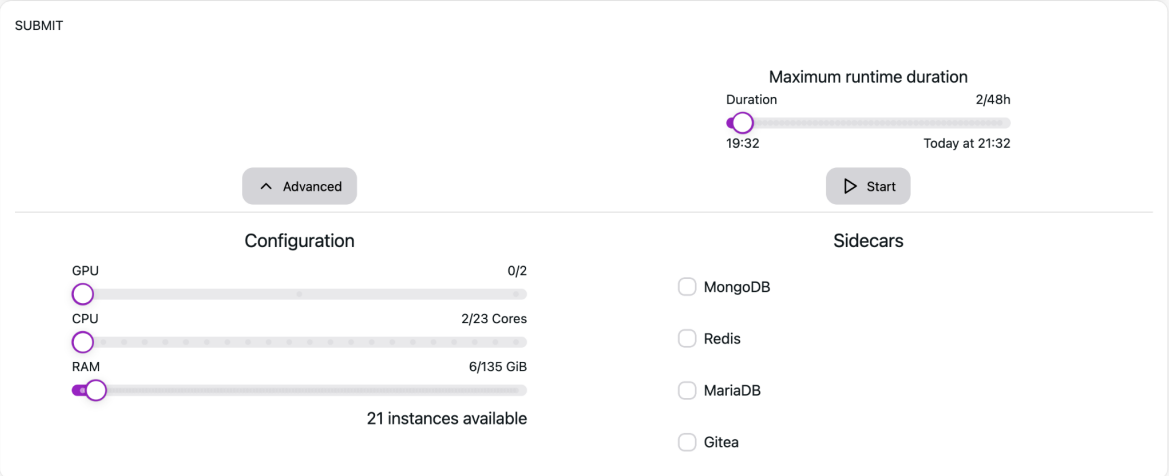


Figure: Advanced setting in Submit Configuration

Click the Advanced button to reveal additional hardware and sidecar configuration options.

Configuration (Hardware Resources)

- GPUs — Select the number of GPUs. Enable GPU acceleration for deep learning, simulations, or rendering workloads.
- CPUs — Set the number of CPU cores. More cores speed up parallelizable computations.
- RAM — Set the amount of RAM. Larger datasets and models require more memory.

Sidecars (Additional Services)

Sidecars are optional companion services that run alongside your main computing session. Enable them as needed:

- MongoDB
- Redis
- MariaDB
- Gitea

2.6 Starting the Server

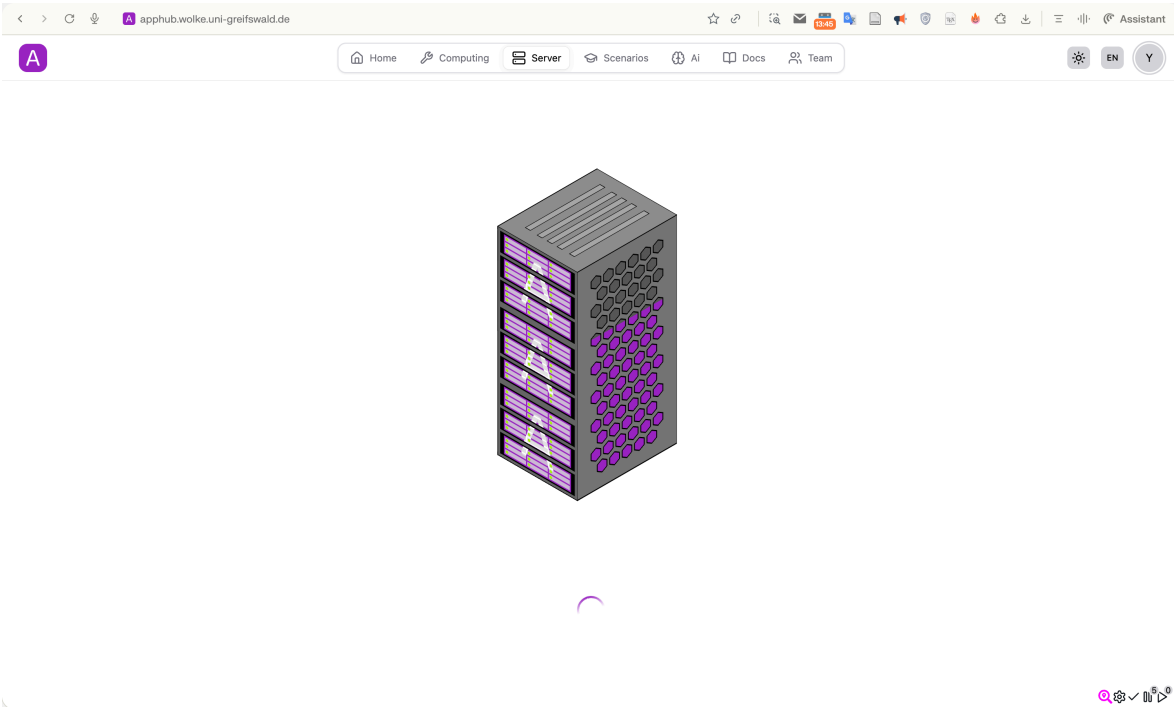


Figure: Starting Server

When you have configured all settings, click the Start button to launch your server. AppHub will:

- Provision your server with the selected software stack
- Set up the chosen interface (JupyterLab, VS Code, RStudio, or Desktop)
- Allocate the specified CPUs, RAM, and GPUs
- Start any enabled sidecar services
- Redirect you to the Server tab where you can monitor and open your session

The server takes some time to start. Once ready, click Open to access your computing environment.

3. Server Management

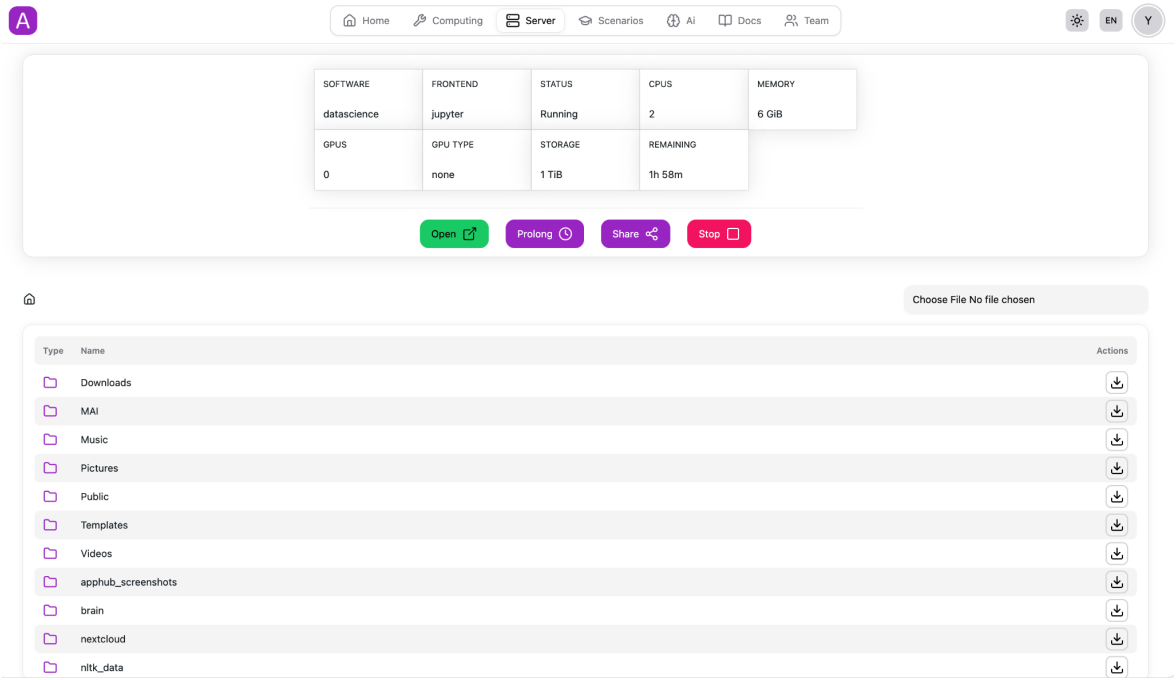


Figure: Server Management

After starting your server, you are automatically taken to the Server tab. This is your control center for managing your active computing session.

The Server page shows:

- Selected Software, Frontend, Server status, CPUs used, Memory, GPUs used, GPU type, Storage and Remaining time.
- Open — Click to open your computing environment in a new browser tab
- Prolong — Extend the runtime of your server if more time is needed
- Share — Share your server session with another user
- Stop — Shut down the server.
- File Browser — Browse, upload, or download files from your server home directory directly in the AppHub interface

Tip: Always save your work before the runtime expires. Use Prolong to add more time without interrupting your session.

4. Scenarios

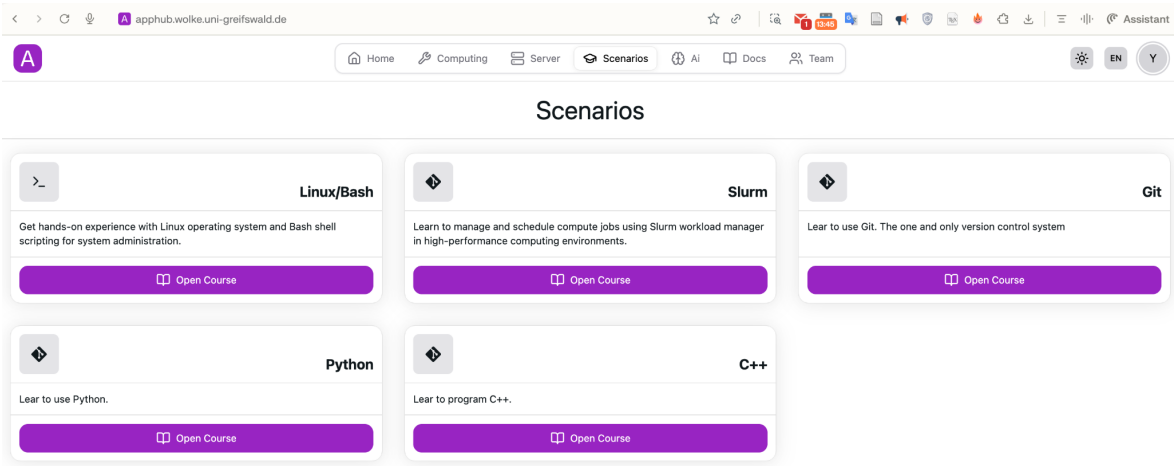


Figure: Scenarios Page — interactive learning modules including Linux/Bash, Slurm, Git, Python, and C++

The Scenarios section provides interactive, hands-on tutorials and learning environments. These guided exercises help new and experienced users understand how to use the tools available on AppHub.

Available Courses

- Linux / Bash — Learn the fundamentals of the Linux command line: file navigation, permissions, scripting, and process management.
- Slurm — Understand the Slurm workload manager for scheduling and running jobs on HPC (High-Performance Computing) clusters.
- Git — Master version control with Git: committing, branching, merging, and collaborating on code repositories.
- Python — Hands-on Python programming.
- C++ — Introduction to C++ programming.

Each scenario opens an interactive playground where you can run commands and code directly in the browser without needing to configure any software.

5. AI Services

The AI section of AppHub provides access to a suite of Artificial Intelligence services designed for scientific and research use. Navigate to the AI tab to explore the available chatbots and AI-powered tools.

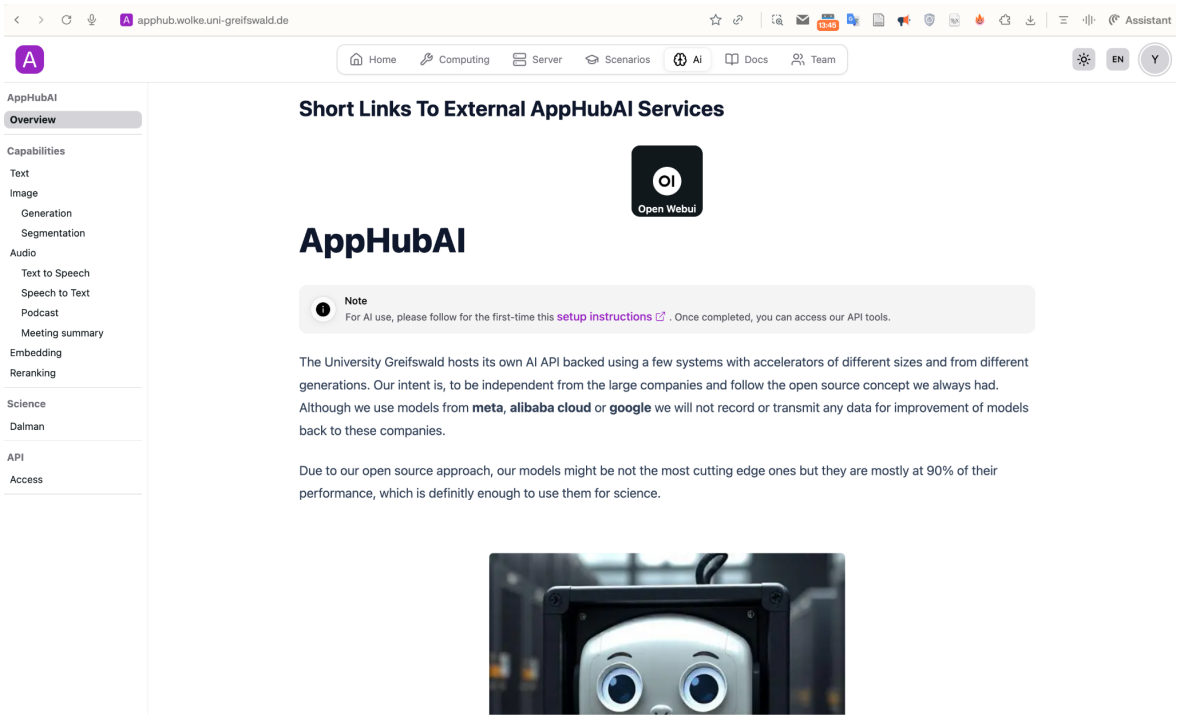


Figure: AI Services Page — overview of available AI models and chatbots

AI Capabilities

- Text (LLM Chat) — Interact with large language models (LLMs) for question answering, text generation, code assistance, and scientific writing.
- Image Generation — Generate images from text prompts using AI image models.
- Image Understanding — Analyze and describe images using vision-language models.
- Audio Transcription — Convert spoken audio to text using speech recognition models.
- Text-to-Speech — Convert written text into natural-sounding audio.
- API Access — All AI capabilities are accessible via the AppHubAI API for integration into your own applications, scripts, and workflows.

These AI services are hosted locally on AppHub infrastructure, ensuring data privacy and compliance with university data protection requirements. No data is sent to external commercial AI providers.

6. Documentation

The Docs section contains official technical documentation, guides, and references for using AppHub and its related services.

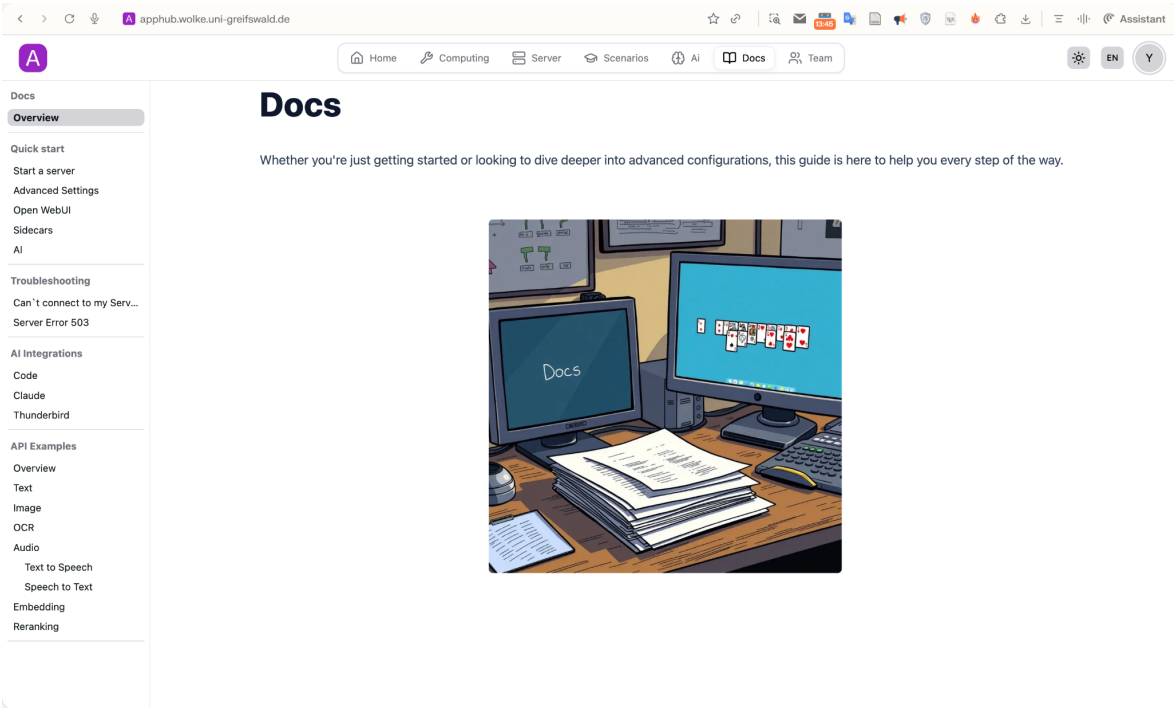


Figure: Documentation Page — technical guides and references for AppHub services

The documentation covers topics including:

- Getting started guides for new users
- API documentation for the AppHubAI services
- Tutorials for specific scientific workflows

7. Development Team

The Team section introduces the people responsible for developing, maintaining, and improving the AppHub platform.

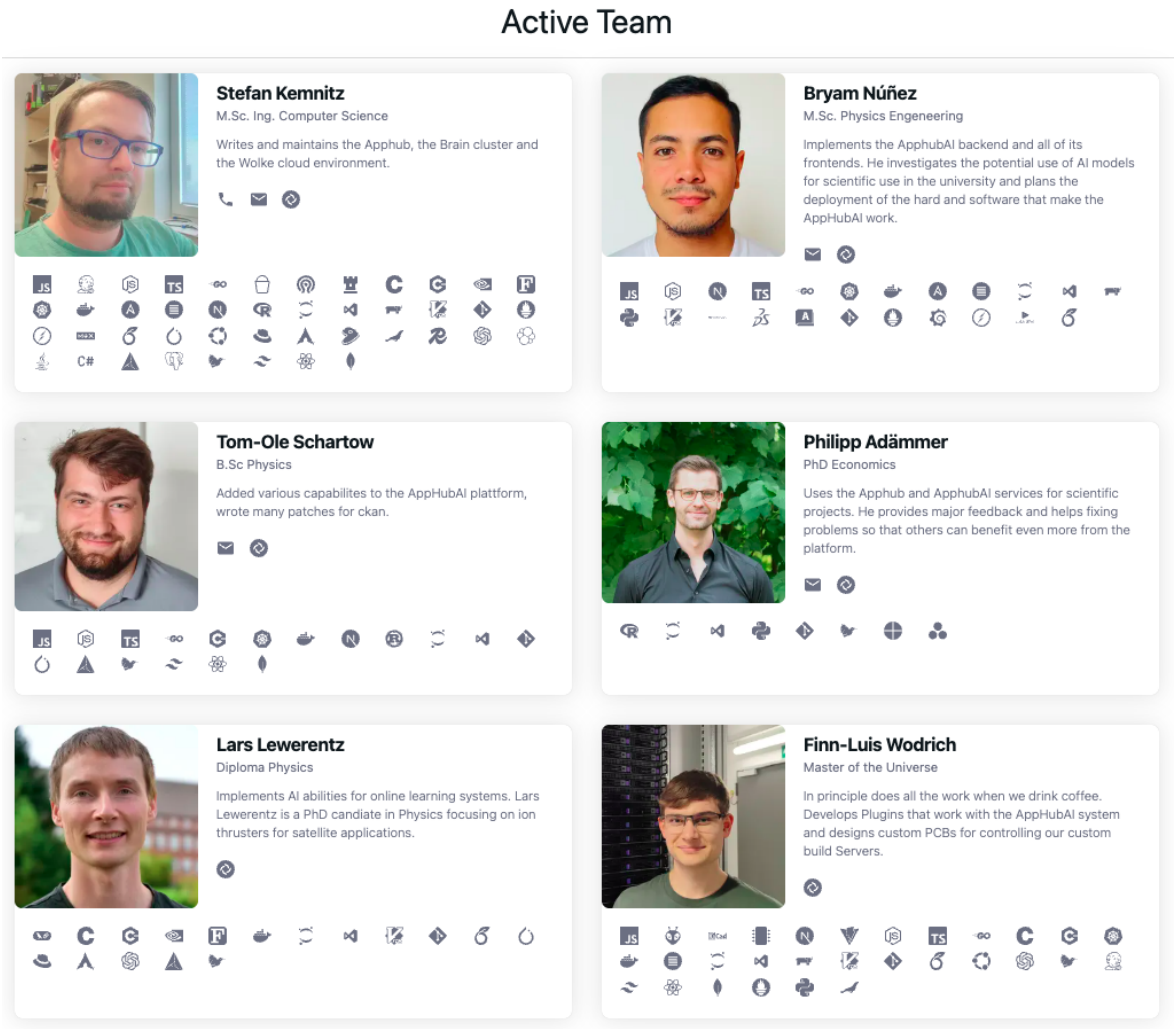


Figure: Team Page — active team members behind AppHub

Active Team Members

- Stefan Kemnitz — Writes and maintains the AppHub, the Brain cluster, and the Wolke cloud environment.
- Bryam Núñez — Implements the AppHubAI backend and frontends; investigates AI models for scientific use and plans hardware/software deployment.
- Tom-Ole Schartow — Adds capabilities to the AppHubAI platform and writes patches for CKAN.
- Philipp Adämmer — Uses AppHub/AppHubAI services for scientific projects and provides feedback to improve the platform.

- Lars Lewerentz — Implements AI abilities for online learning systems; PhD candidate in Physics.
- Finn-Luis Wodrich — Develops plugins for the AppHubAI system and designs custom PCBs for server control.

8. Contact

Dipendra Yadav – dipendra.yadav@uni-greifswald.de