Computational Materials Science 188 (2021) 110165

Contents lists available at ScienceDirect

Computational Materials Science

journal homepage: www.elsevier.com/locate/commatsci

AiiDAlab – an ecosystem for developing, executing, and sharing scientific workflows

Aliaksandr V. Yakutovich [a,b,d,1,*], Kristjan Eimre [c,d,1], Ole Schütt [c,d,1], Leopold Talirz [a,b,d], Carl S. Adorf [b,d], Casper W. Andersen [b,d], Edward Ditler [c,d], Dou Du [b,d], Daniele Passerone [c,d], Berend Smit [a,d], Nicola Marzari [b,d], Giovanni Pizzi [b,d,*], Carlo A. Pignedoli [c,d,*]

[a] Laboratory of Molecular Simulation (LSMO), Institut des Sciences et Ingenierie Chimiques, Valais, École Polytechnique Fédérale de Lausanne, CH-1951 Sion, Switzerland
[b] Theory and Simulation of Materials (THEOS), Faculté des Sciences et Techniques de l'Ingénieur, École Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland
[c] Nanotech@surfaces Laboratory, Swiss Federal Laboratories for Materials Science and Technology (Empa), CH-8600 Dübendorf, Switzerland
[d] National Centre for Computational Design and Discovery of Novel Materials (MARVEL), École Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland

ARTICLE INFO

Keywords:
Web platform
Scientific workflows
Simulations
Data management
Provenance
FAIR data

ABSTRACT

Cloud platforms allow users to execute tasks directly from their web browser and are a key enabling technology not only for commerce but also for computational science. Research software is often developed by scientists with limited experience in (and time for) user interface design, which can make research software difficult to install and use for novices. When combined with the increasing complexity of scientific workflows (involving many steps and software packages), setting up a computational research environment becomes a major entry barrier. AiiDAlab is a web platform that enables computational scientists to package scientific workflows and computational environments and share them with their collaborators and peers. By leveraging the AiiDA workflow manager and its plugin ecosystem, developers get access to a growing range of simulation codes through a python API, coupled with automatic provenance tracking of simulations for full reproducibility. Computational workflows can be bundled together with user-friendly graphical interfaces and made available through the AiiDAlab app store. Being fully compatible with open-science principles, AiiDAlab provides a complete infrastructure for automated workflows and provenance tracking, where incorporating new capabilities becomes intuitive, requiring only Python knowledge.

1. Introduction

Computational simulations constitute an integral part of contemporary science, and are often referred to as the "third pillar of science", after theory and experiments [1]. So far, performing simulations has been considered a task for computational scientists, despite being more and more apparent that simulations, as a fundamental bridge between theory and experiment, can benefit a much larger community. To date, running advanced scientific software requires expert knowledge not only on the phenomena to be modelled, but also on how to access and operate compute resources, develop, compile, and install the relevant software, and how to optimally use high-performance computing (HPC) resources: from the choice of parallelisation strategy to the most efficient number of computational nodes. To mitigate these challenges and to offer a common ground for researches, we introduce AiiDAlab [2,3], an open-source platform that provides the means to develop, execute, and share computational workflows as intuitive web services. Exploiting modern web technologies, AiiDAlab grants the capability of working with advanced simulation tools to any interested researcher.

AiiDAlab is built on top of AiiDA [4–6], an open-source Python infrastructure developed to help researchers automating, managing, persisting, sharing and reproducing workflows and associated data. AiiDA has enabled the community to perform high-throughput simulations [7,8] and publish the full provenance information of all steps performed [9]. In AiiDAlab AiiDA workflows together with user-friendly graphical interfaces for execution and data analysis can be packaged

* Corresponding authors
E-mail addresses: aliaksandr.yakutovich@epfl.ch (A.V. Yakutovich), giovanni.pizzi@epfl.ch (G. Pizzi), carlo.pignedoli@empa.ch (C.A. Pignedoli).
[1] Authors contributed equally.

https://doi.org/10.1016/j.commatsci.2020.110165
Received 29 September 2020; Accepted 2 November 2020
Available online 6 December 2020
0927-0256/© 2020 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

into accessible applications. The platform provides an intuitive environment to manage, run, and publish these applications. AiiDAlab applications are primarily implemented in Python, leveraging Jupyter [10] and IPython widgets [11] for the user interfaces. This enables every computational scientist with Python knowledge to develop and publish applications on the AiiDAlab platform.

The challenges in tackling scientific workflows, data sharing and reproducibility have been addressed by a number of initiatives launched in the last decade. Some of these resulted in the creation of repositories for storage, management, and sharing of scientific data [12–19]. Others focused on developing general formats to store the results computed with different quantum chemistry packages [20,21]. Finally, a consortium of scientists, industry, funding agencies, and scholarly publishers has formulated the "FAIR" principles [22] that are supposed to guide the data holders in making their data reusable.

Platforms providing a web-environment to run scientific software already exist. Notably within industry [23–26] (and thus closed source) or open access [27]. In 2018 Google released to the public Google Colab [28], a Jupyter based collaborative platform.

Here we highlight how AiiDAlab, when coupled with the Materials Cloud [29,30] dissemination platform, can form an integrated solution for seamlessly creating FAIR [22] data. Automation allows greatly accelerating human response as a bottleneck in materials design, which occurs whenever the time needed to correctly setup calculations is comparable to the time it takes to execute them. AiiDAlab allows to easily enable high levels of standardisation in the way calculations are performed and in the way results are analyzed and processed. In a research laboratory where experimentalists collaborate with computational scientists, the platform enables the experimentalists to independently access the submission of calculations and the analysis of the automatically processed data.

This manuscript is divided into two main sections: a detailed description of the AiiDAlab architecture and the presentation of three different AiiDAlab applications.

2. The AiiDAlab platform

The main goal of the AiiDAlab platform is to provide an environment for users with diverse expertise to access and run computational workflows embedded in AiiDAlab apps. The general structure of the platform is shown in Fig. 1. Each user has a separate AiiDAlab account, which grants access to their AiiDAlab instance through a web browser. The AiiDAlab instances are Docker containers running on a remote cloud server with a Jupyter-based environment to manage AiiDAlab apps and an AiiDA instance that takes care of communicating with remote computers, executing automatically all steps defined in the application workflows, and retrieving and parsing the output data. The concepts of the platform are described in detail in the following sections.

2.1. AiiDAlab application

Central to AiiDAlab is the concept of an "application" (app): a set of AiiDA workflows that encode simulation tasks, combined with graphical user interfaces (GUIs) to manage them. The apps allow users to submit calculations to different remote computational facilities, monitor the status of the calculations, analyze the data that are automatically retrieved, and run post-processing calculations.

The core of a scientific AiiDAlab application is constituted by the AiiDA workflows, which encode the computational and scientific knowledge about a simulation. The development of a reliable computational workflow is a complex design problem. The scientists developing the workflow are responsible for the methodologies and related approximations that are selected to tackle the scientific problem under investigation. Additionally, an efficient workflow should be robust and resistant to externalities, such as temporary failures of the remote computing facility, network failures or shutdowns. Several such cases can be resolved by AiiDA itself; for instance, if a computer becomes unavailable, AiiDA detects it and puts the calculation on hold. However, some situations are specific to a given simulation code (run time error due to lack of convergence of an interative routine, or a problem with the input parameters). These cases must be handled by the corresponding workflows.

All user interactions within an AiiDAlab app are facilitated by Jupyter notebooks [10], which are used to submit and manage the AiiDA workflows, and then visualize, analyze and post-process their results. The interactive GUI elements, or "widgets", are mainly built using Python and the ipywidgets module [11]. Development and maintenance of the Jupyter notebooks is straightforward and can be done directly in the Python code. To turn notebooks into user-friendly web applications, a Jupyter extension that we developed, named "Appmode" [38] is enabled by default. This extension hides away all the code from the Jupyter notebooks, leaving only the GUI elements visible to the user.

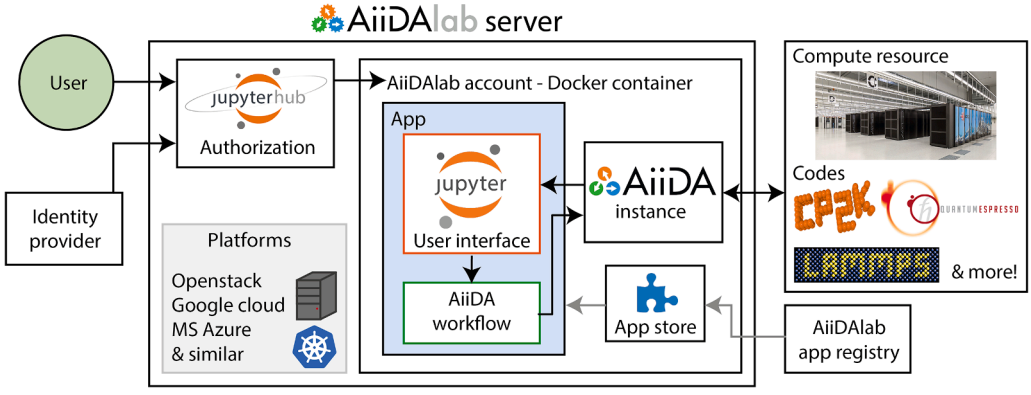

Fig. 1. Diagram of the AiiDAlab architecture. Users are authorized through JupyterHub (which performs the authentication through an external identity provider service) and assigned to a specific personalized AiiDAlab instance – a Docker container with an AiiDA instance and a Jupyter-based environment to manage AiiDAlab apps. The apps contain Jupyter-based user interfaces and AiiDA workflows, which communicate with the AiiDA instance. New apps can be installed through the App store app, which is a user interface to the AiiDAlab application registry. App developers can modify the apps through their AiiDAlab account or directly push their changes to the code repository. The AiiDAlab service can be hosted on a Kubernetes cluster, on a conventional cloud server (OpenStack, Amazon Web Services, Google Compute Cloud, Microsoft Azure, etc), or on a local server. Images and logos are reproduced in compliance to the corresponding licenses [31–37].

As an example, Fig. 2 shows screenshots of the graphical user interfaces of the recently released Scanning Probe Microscopy application, which is designed for the characterization of molecules adsorbed on substrates by means of density functional theory calculations. The app contains two AiiDA workflows: one to simulate scanning tunnelling microscopy (STM) experiments, and one to investigate how the orbitals of a molecule hybridize with the orbitals of the substrate. The submission interfaces (Fig. 2a,c) guide the user in specifying a minimal set of input parameters needed to prepare and submit the workflows. Furthermore, the app includes interfaces to visualize and export the results that have been automatically post-processed, shown in Fig. 2b,d. This Scanning Probe Microscopy app is discussed in more detail in Section 3.3.

For more technical details about AiiDAlab applications, the reader is referred to the "AiiDAlab application" section of the Supplementary Information, with information about the structure of the applications, their creation, and examples of usage of Appmode.

Importantly, a library of user-interface elements (widgets) and their logic to interface with the underlying AiiDA infrastructure is available as a base app called "AiiDAlab widgets". This app provides out-of-the-box some of the high-level functionalities that are most often needed in AiiDAlab applications. Examples of such high-level functionalities include uploading data from a local computer, browsing data that are stored in an AiiDA database, and selecting the computational resources where to run the simulations (see more in the "Monitoring submitted calculations" section of the Supplementary Information). This base app is contained in the GitHub repository aiidalab-widgets-base [39]; every AiiDAlab user can easily import this module and start using the base widgets capabilities. An extended documentation containing the available widgets and examples of usage are available in a dedicated webpage [40].

2.2. AiiDAlab home app

The **AiiDAlab home app** [41] is the core application of the service. It takes care of displaying the Home page (Fig. 3a) and and enables basic interactions with the AiiDAlab machine. At the top of the page, it contains five buttons for quick access to handy tools: *File Manager* to browse files, *Terminal* to access a Linux terminal directly from the browser, *Tasks* to list all running AiiDAlab apps, *App Store* to install new applications and *Help* referencing to the AiiDAlab documentation. *App Store* is a user-friendly interface to the Git repositories of the available apps, which are listed on the AiiDAlab registry (see Section 2.3); it allows to install, uninstall, and update apps with just one click (Fig. 3b). When installing an app, the user can choose between its various versions defined by the author in the corresponding Git repository (Fig. 3c).

2.3. Publishing AiiDAlab applications

Available AiiDAlab apps are listed in a central "AiiDAlab registry" [42]. Once an app is added to the registry, it is automatically listed in the AiiDAlab App Store of every AiiDAlab user. In order for this to work, the app must be in a repository of the Git [43] version-control system accessible by AiiDAlab, like e.g. on a GitHub or GitLab server [44,45]. Each entry in the registry contains the minimum information about an app: its name, a link to its Git repository, a link to a metadata file, and a list of categories it falls within. Such an infrastructure provides AiiDAlab application developers an efficient way to share their scientific expertise and knowledge (encoded into the application and the workflows) with the rest of the community, while maintaining full ownership and control on the development of the apps. In addition, AiiDAlab allows to share simulation data with their whole provenance (i.e. the history of how each item has been generated, by which calculation and with which input) thanks to the use of AiiDA as the underlying workflow engine.

2.4. Distribution and availability of the AiiDAlab service

We provide two deployment solutions for AiiDAlab: a "distributed model" and a "centralized model", that are described below. From the interface point of view, both solutions are identical.

The distributed model follows the traditional way of sharing software. To share an entire stack, we provide a Virtual Machine (VM) image for download, which users can run locally. This is the case of Quantum Mobile [46] – a VirtualBox [47] image that includes AiiDAlab pre-configured. The advantage of this approach is that users retain full control of their own data, and can run on their computers without the need of a central infrastructure or the need of creating login credentials in an online system. The downside is that users have to take responsibility for most of the administrative work, such as backup and updates. Another disadvantage of the distributed model is that there is no control on which VM image version a specific user will be working on, which requires additional work to make sure that apps remain portable.

The centralized model is generally referred to as Software-as-a-Service (SaaS) or Platform-as-a-Service (PaaS). To date we provide the following centralized AiiDAlab platforms as a service [2]: one on cloud servers hosted at the Swiss Supercomputing Center (CSCS), and one on the CESNET servers in Czech Republic (with resources granted by EOSC–hub)[2], both providing high-scalability features via Kubernetes. This approach can be replicated elsewhere straightforwardly. The centralized model lowers the entrance barrier for a user, since only login credentials and a web browser are required to access the cloud service, without the need to install custom software. At the same time, however, this model requires web engineers to administer and maintain the service. In order to work in Kubernetes [48], AiiDAlab is distributed also within a docker container. We emphasise that the same AiiDAlab docker container which runs on Kubernetes can also be run locally if the user wants to. The necessary setup scripts are provided in the corresponding repository [49]. More details about the AiiDAlab docker container are provided in the "AiiDAlab docker stack and AiiDAlab package" section of the Supplementary Information.

Last, it should be noted that companies could have strict policies on the physical location of their data, often requiring "in house" storage capabilities. For this reason, we have decided to license all core parts of AiiDAlab with the industry-friendly MIT open source licence [50] allowing for companies to deploy a custom AiiDAlab server on their premises for internal use. Moreover, to facilitate re-deployment and setting up a (new) AiiDAlab instance, we provide automated scripts to re-build the whole infrastructure using the Ansible tool [51]. Specifically, two Ansible roles are provided: ansible-role-aiidalab [52] that installs the AiiDAlab environment on Linux-like operating systems, and ansible-role-aiidalab-server [53] that deploys a multi-user AiiDAlab server on Ubuntu. Additionally, to set up an automatically scaling platform, we provide step-by-step installation instructions [54] to deploy AiiDAlab on a Kubernetes [48] cluster.

3. AiiDAlab use cases

In order to showcase the benefits of the AiiDAlab platform, we describe some use cases in which it has already been used in production, driven by the needs of experimental research in on-surface chemistry. On-surface chemistry is a powerful method toward bottom-up fabrication of complex nanostructures, unattainable via traditional solution chemistry [55]. In this field, the nanotech@surfaces laboratory at Empa combines scanning probe microscopy techniques and atomistic simulations to study polymerization of molecules on catalytic surfaces. In 2010 the laboratory demonstrated a bottom-up approach [56] for the

[2] The EOSC–hub instance is connected to European Grid Infrastructure (EGI) check-in for authentication, giving access to a wide community of researchers.

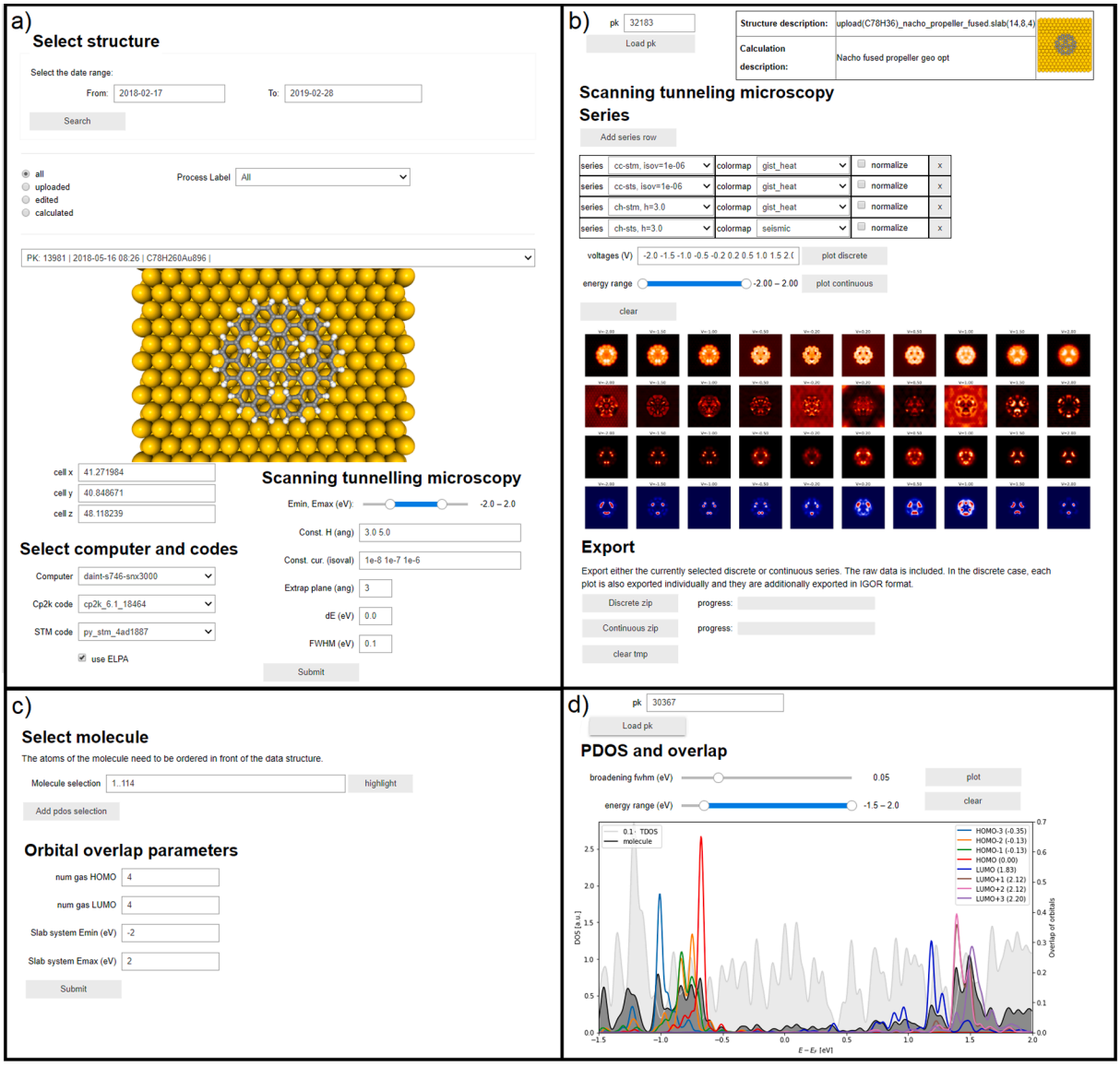

Fig. 2. Snapshots of the Scanning Probe Microscopy (SPM) app. (a) Interface for the submission of a workflow to compute Scanning Tunnelling Microscopy (STM) images. (b) Interface to analyze the results of the STM calculations. (c) Interface to submit the workflow to compute the Projected Density of States (PDOS) for a system composed by a molecule adsorbed on a substrate. (d) Analysis of the PDOS results. See Fig. S10 for more details.

synthesis of graphene nanoribbons (GNRs) and since then a number of GNRs with different topologies [57] have been fabricated with atomic precision. To understand the fabrication process and to predict the electronic properties of the nanomaterials, state-of-the-art computational methods have to be applied routinely with protocols that have been optimized over the last ten years. A bottleneck in the discovery process then becomes the "human factor": the time needed to manually set up calculations, process them, and identify and correct human errors. The classes of calculations that the laboratory deals with include electronic-structure characterization of gas-phase GNRs, identification of intermediate states and activation barriers for the chemical reactions involving molecular precursors adsorbed on noble metal substrates, and on-surface scanning probe microscopy and spectroscopy.

Our goal then has been to create AiiDAlab applications with intuitive user interfaces to perform automated workflows for the calculations mentioned above. We designed the apps in such a way so that they could be used also by researchers who are not familiar with the details needed to set up a calculation.

3.1. Nanoribbon app

Obtaining a specific GNR structure requires a considerable effort from the experimental point of view, both in terms of solution chemistry processes needed to fabricate the appropriate molecular precursors and in terms of the "on-surface synthesis" itself. Key steps before entering the experimental effort are the prediction of the relevant electronic properties associated to GNRs with a specific geometry and understanding how their topology and composition (for example insertion of heteroatoms) allow to tune such properties. The Nanoribbon AiiDAlab app [58] has been designed, in direct collaboration with experimentalists, to streamline these tasks.

The app provides a user friendly interface to run the automatic characterization of one-dimensional nanostructures using the Quantum ESPRESSO distribution [59]. The calculation steps involve cell and geometry optimizations, band structure calculations, and exports of various electronic structure properties. Overview of the various user interfaces and calculation steps is shown as a flowchart in Fig. 4. Additional details about the Nanoribbon app can be found in the relevant section of the Supplementary Information.

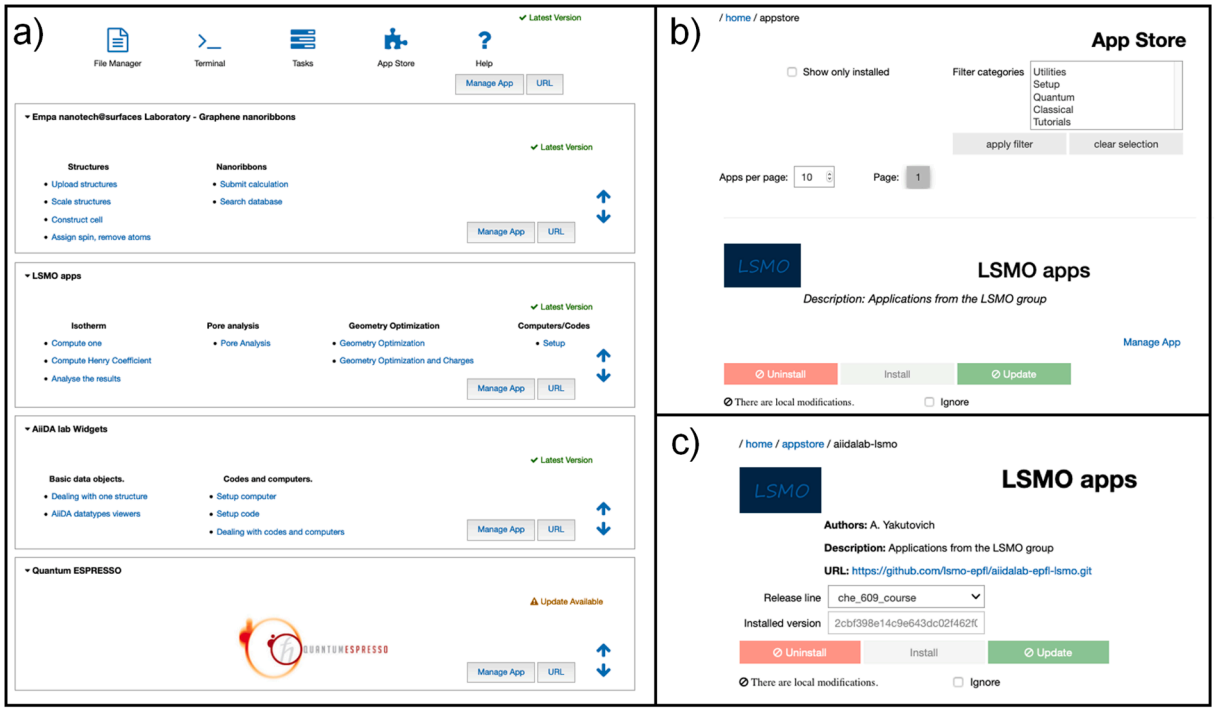

Fig. 3. (a) Snapshot of the AiiDAlab starting page. At the top, the links to five home applications are visible. Below, additional panels are present for each of the apps installed via the App Store. The viewing of this list can be customized, and in this example includes (among others) the Nanoribbon app discussed in Section 3.1. (b) Snapshot of the App Store application. The user can select the number of apps shown per page, filter by application groups, and choose whether to show all apps or only the installed ones. (c) Snapshot of the App management page that provides more information about the chosen app and allows selection of a s.pecific version.

3.2. On-surface chemistry app

To deal efficiently with the computational tasks needed to interpret and predict the outcome of experiments related to on-surface chemistry, the On-Surface Chemistry AiiDAlab app [60] allows to run a variety of first-principles calculations at multiple levels of theory, using the CP2K simulation package [61].

The app provides workflows to run geometry optimizations and nudged-elastic-band (NEB) calculations [62,63] to identify reaction paths. Additionally, a workflow is provided to compute a physically feasible initial path for a NEB calculation using a chain of constrained geometry optimizations: starting from an optimized geometry, the user defines a collective variable (CV) (e.g. a bond length) and a list of values that the CV should fulfill in sequence. After the geometry of the current step reaches equilibrium, fulfilling the current constraint for the CV, a new optimization starts imposing the next constraint value in the list. This procedure allows to identify a reasonable guess for the reaction path when linear interpolation between an initial and a final state would result in a non-physical path. Several supporting tools are provided as Jupyter notebooks, allowing for instance to modify structural geometry, create metal slabs, and conveniently access and export the calculation results directly from the browser. An overview of the various user interfaces and calculation steps is shown as a flowchart in Fig. 5. Additional details about the On-Surface Chemistry app can be found in the Supplementary Information.

3.3. Scanning probe microscopy app

Scanning probe microscopy is the main experimental tool to acquire structural and electronic information about molecules and nanostructures on metal surfaces. More specifically, scanning tunnelling microscopy (STM) and spectroscopy (STS) allow to determine the spatial distributions of the electronic states and the electronic band gap of the system under investigation. However, to validate the correctness of the assignment of the electronic states, to investigate the effect of the substrate, and to determine the geometry of the system, experimental electronic signatures need to be compared with computational simulations. To run these simulations, we have developed the Scanning Probe Microscopy app [64]. The app includes a workflow for the simulation of STM/STS images and a workflow to run projected density of states (PDOS) analysis together with orbital hybridization investigation for molecules adsorbed on metal slabs. An overview of the various user interfaces and calculation steps is shown as a flowchart in Fig. 6.

The interfaces for submission and for analysis of the result for the two workflows provided in the Scanning Probe Microscopy app are shown in Fig. 2. In the submission interface of the STM/STS workflow (Fig. 2a), the user can specify the input geometry from the AiiDA database, the remote computer and codes, and the STM parameters. The corresponding viewing interface, shown in Fig. 2b, allows to visualize a series of images by specifying discrete bias voltage values or a continuous range of energies. In the example shown, a discrete series of images are visualized for a porous nanographene molecule. Additionally, the selected visualizations can be exported as a ZIP file archive containing the images in PNG, IGOR Pro and raw data formats. The orbital hybridization workflow submission interface contains the same geometry and code selection widgets as the STM/STS workflow with the additional widgets illustrated in Fig. 2c. The corresponding viewing interface, shown in Fig. 2d, allows to inspect the PDOS and the orbital overlaps with an effective interactive plot that combines the two. Further details about the Scanning Probe Microscopy app are presented in the Supplementary Information.

4. Conclusions and outlook

We presented the overall concept and the implementation of AiiDAlab, an ecosystem for the development, sharing and deployment of user-friendly simulation apps that 1) provides convenient user interfaces to computational workflows, hiding their technical details and making them straightforwardly available to a broad range of researchers, and 2) facilitates the development of such interfaces via a Python-based

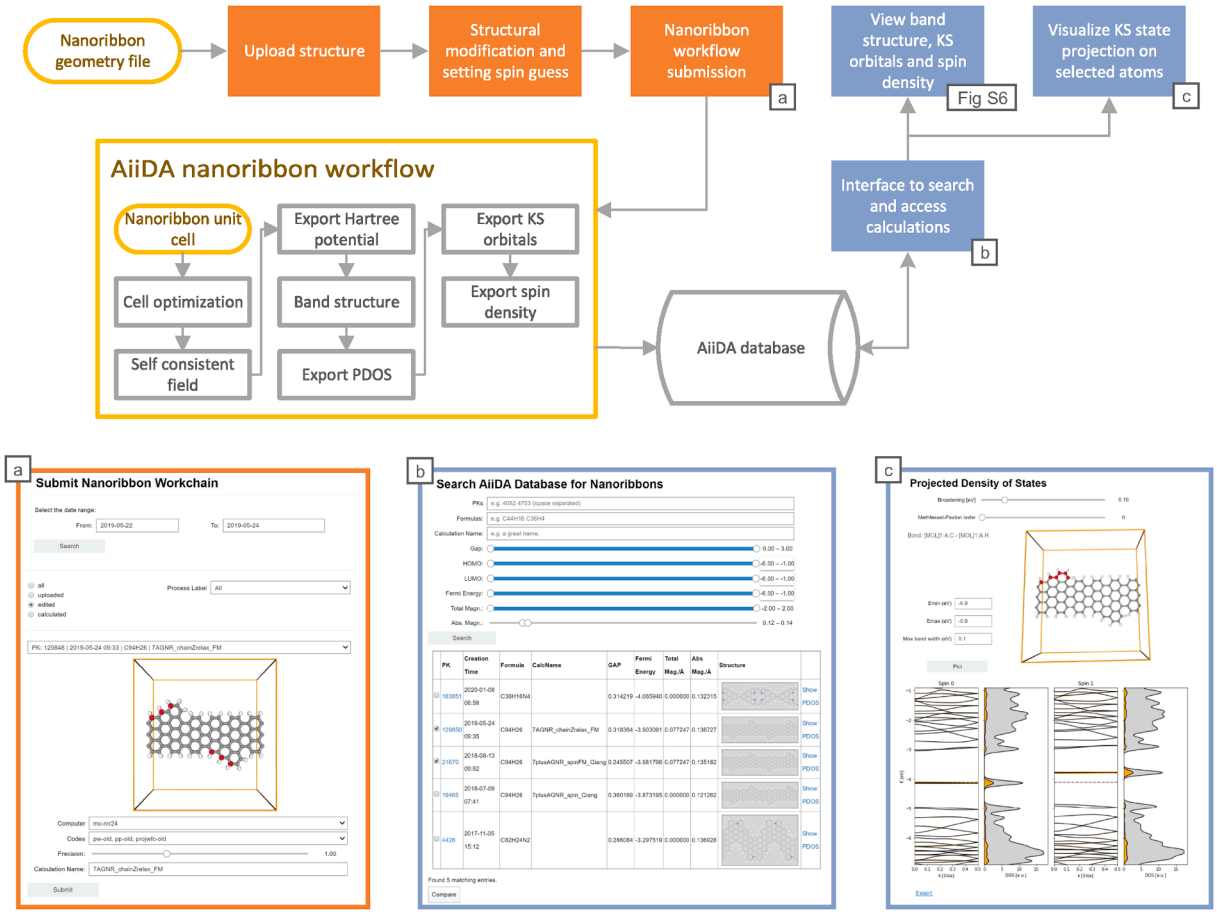

Fig. 4. Conceptual flowchart of the Nanoribbon app together with three screenshots of user interfaces, shown in panels (a), (b) and (c). Oval shapes represent starting points and filled rectangles represent Jupyter user interfaces for the various steps of preparation (orange) and results (blue). A selection of the user interfaces can be seen in the figures/panels referenced in the bottom right corner of the corresponding rectangle. Arrows highlight logical precedence, where the output of a preceding step becomes the input to the next one, together with additional manual input, if necessary. The AiiDA Nanoribbon workflow sub-chart shows an overview of the calculation steps that the AiiDA daemon subsequently submits to the computational resource. Each step is stored in the AiiDA database.

programming language and the availability of common interactive widgets.

AiiDAlab offers the possibility to contribute and share computational workflows and data: the software stack is standardized using Docker containers, which provide every user with the same identical software environment. In order to minimize the entrance barrier for new users, the platform can run directly "in the cloud", requiring only a web browser and a login to access it (servers are currently hosted at CSCS, Switzerland and CESNET, Czech Republic), but can also be redeployed on internal resources or run locally using the Quantum Mobile virtual machine [46]. AiiDAlab leverages the flexibility of the Python language and the power of Jupyter Notebooks to expose a simple programming interface to app developers. Additionally, in order to provide an intuitive user interface, a novel Jupyter extension called Appmode has been developed. Together with Jupyter and the use of widgets (both existing ones, and new ones developed by us), these tools make it possible to quickly create web applications in Python, taking advantage of existing libraries like ipywidgets [11], matplotlib [65], bqplot [66], nglview [67], and ase [68]. Since Python knowledge is widespread in the community, we expect that many scientists will be able to contribute new apps.

We also provide examples of a full range of apps which were developed in collaboration with experimentalists to tackle specific problems in the field of carbon-based nanostructures and surface-supported chemical reactions, enabling to run complex simulations effortlessly.

In addition to the applications presented here, that demonstrate the benefits and functionality of the platform, numerous more apps are currently under development. To facilitate this process, we provide a set of tools for the creation of new apps, including a "cookie cutter" recipe that automates the initial steps of the app creation and the AiiDAlab widgets which can be readily embedded into new apps (see Supplementary Information).

Last, in order to encourage not only the sharing of data, but also the sharing of codes, workflows and applications within the community, we deployed an open AiiDAlab registry. Developers can use it to register their apps and make them automatically available to all other users via the App Store.

AiiDAlab is a modular and efficient platform granting a collaborative environment centered on simulations and targeted to people with different expertise. The platform can serve a broad range of communities beyond the original focus on materials science, and facilitates sharing of turn-key computational workflows and of the data produced by simulations. AiiDAlab allows for a direct exchange of processed simulation results in an appropriate format, introduces an easy way to standardize routine calculations, reducing to a minimum bottlenecks such as human mistakes. We expect that computational scientists from diverse fields will start adopting the platform and contributing to it with workflows and interfaces, empowering the scientific community by making simulation tools more directly and easily accessible to a broader range of researchers.

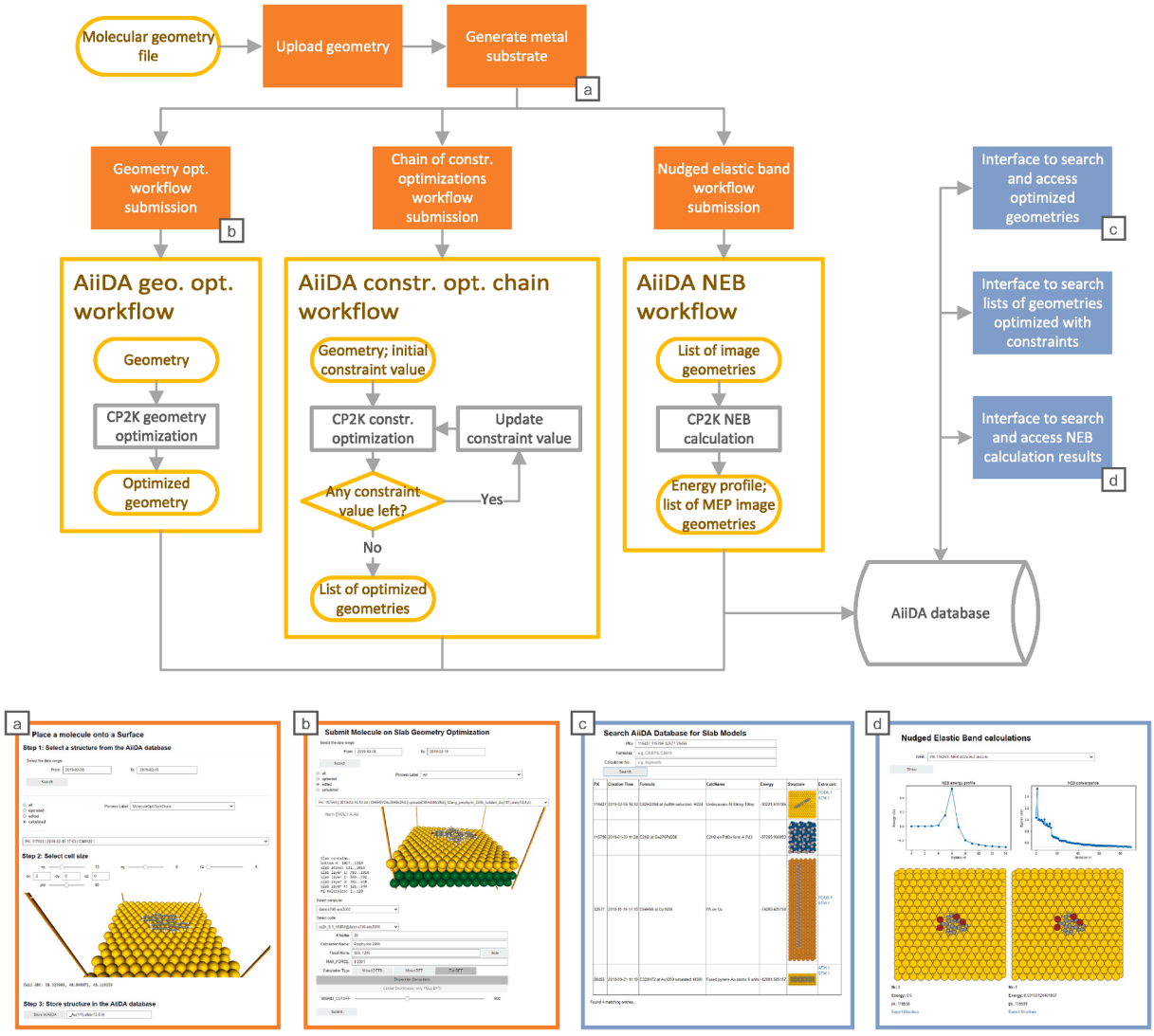

Fig. 5. Conceptual flowchart of the On-Surface Chemistry app together with four screenshots of user interfaces, shown in panels (a), (b), (c) and (d). See the caption of Fig. 4 for the meaning of the various flowchart elements.

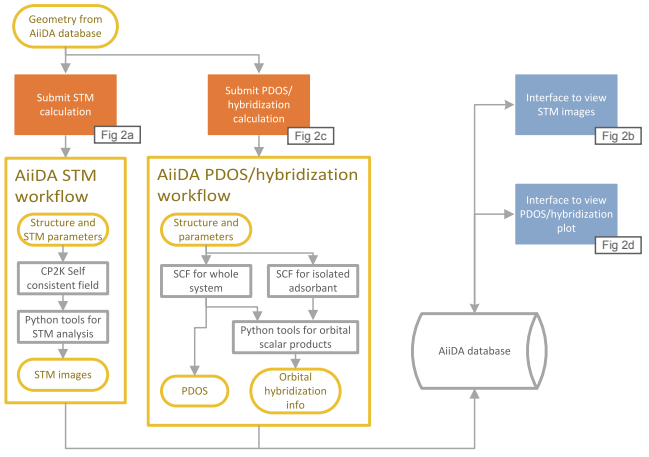

Fig. 6. Conceptual flowchart of the Scanning Probe Microscopy app. See the caption of Fig. 4 for the meaning of the various flowchart elements.

Data Availability

All the code of the AiiDAlab platform as well as the AiiDAlab applications discussed in the paper are open source (released under the MIT license) and can be downloaded from the links provided in the References section.

CRediT authorship contribution statement

Aliaksandr V. Yakutovich: Conceptualization, Methodology, Software, Writing - original draft, Writing - review & editing, Visualization, Supervision. **Kristjan Eimre:** Conceptualization, Methodology, Software, Writing - original draft, Writing - review & editing, Visualization. **Ole Schütt:** Conceptualization, Methodology, Software, Writing - original draft. **Leopold Talirz:** Conceptualization, Software, Investigation, Project administration, Writing - review & editing. **Carl S. Adorf:** Conceptualization, Methodology, Software, Writing - review & editing, Visualization, Supervision. **Casper W. Andersen:** Conceptualization, Software, Investigation. **Edward Ditler:** Software, Investigation. **Dou Du:** Software. **Daniele Passerone:** Conceptualization, Funding acquisition, Writing - review & editing. **Berend Smit:** Project administration, Funding acquisition, Writing - review & editing. **Nicola Marzari:** Project administration, Funding acquisition, Writing - review & editing.

Giovanni Pizzi: Conceptualization, Methodology, Software, Validation, Investigation, Project administration, Funding acquisition, Writing - review & editing, Supervision, Project administration, Funding acquisition. **Carlo A. Pignedoli:** Conceptualization, Validation, Writing - original draft, Writing - review & editing, Visualization, Supervision, Project administration, Funding acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is supported by the MARVEL National Centre for Competency in Research funded by the Swiss National Science Foundation (grant agreement ID 51NF40-182892), the European Centre of Excellence MaX "Materials design at the Exascale" (Grant No. 824143), the "MaGic" project of the European Research Council (grant agreement ID 666983), the swissuniversities P-5 "Materials Cloud" project (grant agreement ID 182-008), the "MARKETPLACE" H2020 project (grant agreement ID 760173), the "INTERSECT" H2020 project (grant agreement ID 814487), the "EMMC" H2020 project (grant agreement ID 723867), the "OSSCAR" project (funded by the EPFL Open Science Fund), and the Swiss National Science Foundation grant agreement ID 172527. We acknowledge PRACE for awarding us simulation time on Piz Daint at CSCS (project ID 2016153543) and Marconi at CINECA (project ID 2016163963), and the support of Swiss Platform for Advanced Scientific Computing PASC.

We thank the CSCS support team for continued support during the deployment of AiiDAlab on their infrastructure. This work used the EGI infrastructure with the dedicated support of CESNET-MCC. We thank the EOSC–Hub Early Adopter Programme for providing access to kubernetes resources.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.commatsci.2020.110165.

References

[1] B. Skuse, Physics World, 32, IOP Publishing, 2019, pp. 40–43, https://doi.org/10.1088/2058-7058/32/3/33.
[2] AiiDAlab, URL: https://www.materialscloud.org/aiidalab.
[3] AiiDAlab documentation, URL: https://aiidalab.readthedocs.io/en/latest/.
[4] G. Pizzi, A. Cepellotti, R. Sabatini, N. Marzari, B. Kozinsky, AiiDA: automated interactive infrastructure and database for computational science, Comput. Mater. Sci. 111 (Supplement C) (2016) 218–230, https://doi.org/10.1016/j.commatsci.2015.09.013.
[5] S. P. Huber, S. Zoupanos, M. Uhrin, L. Talirz, L. Kahle, R. Häuselmann, D. Gresch, T. Müller, A. V. Yakutovich, C. W. Andersen, F. F. Ramirez, C. S. Adorf, F. Gargiulo, S. Kumbhar, E. Passaro, C. Johnston, A. Merkys, A. Cepellotti, N. Mounet, N. Marzari, B. Kozinsky, G. Pizzi, AiiDA 1.0, a scalable computational infrastructure for automated reproducible workflows and data provenance, Scientific Data 7 (1) (2020) 300, number: 1 Publisher: Nature Publishing Group. doi:10.1038/s41597-020-00638-4.
[6] AiiDA – Science, URL: http://www.aiida.net/science/.
[7] N. Mounet, M. Gibertini, P. Schwaller, D. Campi, A. Merkys, A. Marrazzo, T. Sohier, I.E. Castelli, A. Cepellotti, G. Pizzi, N. Marzari, Two-dimensional materials from high-throughput computational exfoliation of experimentally known compounds, Nat. Nanotechnol. 13 (3) (2018) 246–252, https://doi.org/10.1038/s41565-017-0035-5.
[8] AiiDA, URL: http://www.aiida.net/.
[9] N. Mounet, M. Gibertini, P. Schwaller, D. Campi, A. Merkys, A. Marrazzo, T. Sohier, I. E. Castelli, A. Cepellotti, G. Pizzi, N. Marzari, Two-dimensional materials from high-throughput computational exfoliation of experimentally known compoundsType: dataset. doi:10.24435/materialscloud:2017.0008/v3.
[10] Jupyter, URL: https://www.jupyter.org.
[11] ipywidgets, URL: https://github.com/jupyter-widgets/ipywidgets.
[12] P. Villars, M. Berndt, K. Brandenburg, K. Cenzual, J. Daams, F. Hulliger, T. Massalski, H. Okamoto, K. Osaki, A. Prince, H. Putz, S. Iwata, The Pauling File, Binaries Edition, Journal of Alloys and Compounds 367 (1) (2004) 293–297. doi:10.1016/j.jallcom.2003.08.058.
[13] N. Zarkevich, Structural database for reducing cost in materials design and complexity of multiscale computations, Complexity 11 (4) (2006) 36–42, https://doi.org/10.1002/cplx.20117.
[14] A. Jain, G. Hautier, C.J. Moore, S. Ping Ong, C.C. Fischer, T. Mueller, K.A. Persson, G. Ceder, A high-throughput infrastructure for density functional theory calculations, Comput. Mater. Sci. 50 (8) (2011) 2295–2310, https://doi.org/10.1016/j.commatsci.2011.02.023.
[15] S. Curtarolo, W. Setyawan, S. Wang, J. Xue, K. Yang, R.H. Taylor, L.J. Nelson, G.L. W. Hart, S. Sanvito, M. Buongiorno-Nardelli, N. Mingo, O. Levy, AFLOWLIB.ORG: A distributed materials properties repository from high-throughput ab initio calculations, Comput. Mater. Sci. 58 (2012) 227–235, https://doi.org/10.1016/j.commatsci.2012.02.002.
[16] J.E. Saal, S. Kirklin, M. Aykol, B. Meredig, C. Wolverton, Materials Design and Discovery with High-Throughput Density Functional Theory: The Open Quantum Materials Database (OQMD), J. Minerals, Metals Mater. Soc. 65 (11) (2013) 1501–1509, https://doi.org/10.1007/s11837-013-0755-4.
[17] D.D. Landis, J.S. Hummelshøj, S. Nestorov, J. Greeley, M. Dułak, T. Bligaard, J. K. Nørskov, K.W. Jacobsen, The Computational Materials Repository, Comput. Sci. Eng. 14 (6) (2012) 51–57, https://doi.org/10.1109/MCSE.2012.16.
[18] EUDAT - Research Data Services, Expertise & Technology Solutions, URL: https://www.eudat.eu/.
[19] NOMAD Repository, URL: http://www.nomad-repository.eu/.
[20] S. Adams, P.d. Castro, P. Echenique, J. Estrada, M.D. Hanwell, P. Murray-Rust, P. Sherwood, J. Thomas, J. Townsend, The Quixote project: Collaborative and Open Quantum Chemistry data management in the Internet age, J. Cheminf. 3 (1) (2011) 38, https://doi.org/10.1186/1758-2946-3-38.
[21] The Open Provenance Model, URL: https://openprovenance.org/opm/.
[22] M.D. Wilkinson, M. Dumontier, I.J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L.B. da Silva Santos, P.E. Bourne, J. Bouwman, A. J. Brookes, T. Clark, M. Crosas, I. Dillo, O. Dumon, S. Edmunds, C.T. Evelo, R. Finkers, A. Gonzalez-Beltran, A.J.G. Gray, P. Groth, C. Goble, J.S. Grethe, J. Heringa, P.A.C. 't Hoen, R. Hooft, T. Kuhn, R. Kok, J. Kok, S.J. Lusher, M. E. Martone, A. Mons, A.L. Packer, B. Persson, P. Rocca-Serra, M. Roos, R. van Schaik, S.-A. Sansone, E. Schultes, T. Sengstag, T. Slater, G. Strawn, M.A. Swertz, M. Thompson, J.E. van der Leivan Mulligen, J. Velterop, A. Waagmeester, P. Wittenburg, K. Wolstencroft, J. Zhao, B. Mons, The FAIR Guiding Principles for scientific data management and stewardship, Sci. Data 3 (2016) 160018, https://doi.org/10.1038/sdata.2016.18.
[23] L. Koschmieder, S. Hojda, M. Apel, R. Altenfeld, Y. Bami, C. Haase, M. Lin, A. Vuppala, G. Hirt, G. Schmitz, AixViPMaP—-an Operational Platform for Microstructure Modeling Workflows, Integrating Mater. Manuf. Innovation 8 (2) (2019) 122–143, https://doi.org/10.1007/s40192-019-00138-3.
[24] MedeA Software, URL: https://www.materialsdesign.com/medea-software.
[25] Citrine Informatics, URL: https://citrine.io/.
[26] Exabyte.io - Materials Discovery Cloud, URL: https://exabyte.io/.
[27] nanoHUB.org - Simulation, Education, and Community for Nanotechnology, URL: https://nanohub.org/.
[28] Google Colaboratory, URL: https://colab.research.google.com/notebooks/intro.ipynb.
[29] L. Talirz, S. Kumbhar, E. Passaro, A. V. Yakutovich, V. Granata, F. Gargiulo, M. Borelli, M. Uhrin, S. P. Huber, S. Zoupanos, C. S. Adorf, C. W. Andersen, O. Schütt, C. A. Pignedoli, D. Passerone, J. VandeVondele, T. C. Schulthess, B. Smit, G. Pizzi, N. Marzari, Materials Cloud, a platform for open computational science, Scientific Data 7 (1) (2020) 299, number: 1 Publisher: Nature Publishing Group. doi:10.1038/s41597-020-00637-5.
[30] Materials Cloud, URL: https://www.materialscloud.org/.
[31] CSCS supercomputer image, license: CC BY-SA 3.0, URL: https://www.cscs.ch/galleries/supercomputers/11_supercomputer.jpg.
[32] CP2K logo, license: CC BY-SA 4.0, URL: https://www.cp2k.org/logo.
[33] Quantum Espresso logo, license: GPL 2.0, URL: https://www.quantum-espresso.org/project/logos/Quantum_espresso_logo.jpg.
[34] LAMMPS logo, license: GPL 2.0, URL: https://github.com/lammps/lammps/blob/master/doc/utils/sphinx-config/_static/lammps-logo.png.
[35] Jupyter logo, license: BSD 3-Clause, URL: https://github.com/jupyter/jupyter.github.io/blob/master/assets/main-logo.svg.
[36] Jupyterhub logo, license: BSD 3-Clause, URL: https://jupyter.org/assets/hublogo.svg.
[37] Kubernetes logo, license: Apache-2.0, URL: https://github.com/kubernetes/kubernetes/blob/master/logo/logo.svg.
[38] Appmode: a Jupyter extension that turns notebooks into web applications, URL https://github.com/oschuett/appmode.
[39] Reusable widgets for AiiDAlab applications, URL: https://github.com/aiidalab/aiidalab-widgets-base.
[40] AiiDAlab widgets base documentation, URL: https://aiidalab-widgets-base.readthedocs.io/en/latest/.
[41] AiiDAlab Home App, URL: https://github.com/aiidalab/aiidalab-home.
[42] AiiDAlab registry, URL: https://github.com/aiidalab/aiidalab-registry.
[43] Git, URL: https://www.git-scm.com/.
[44] GitHub, URL: https://github.com.
[45] GitLab. URL: https://about.gitlab.com/.
[46] Quantum Mobile, URL: https://www.materialscloud.org/work/quantum-mobile.
[47] Oracle VM VirtualBox. https://www.virtualbox.org/.
[48] Production-Grade Container Orchestration - Kubernetes, URL: https://kubernetes.io/.

[49] The Docker software stack for the AiiDAlab, URL: https://github.com/aiidalab/aiidalab-docker-stack.
[50] The MIT License, URL: https://opensource.org/licenses/MIT.
[51] Ansible is Simple IT Automation, URL: https://www.ansible.com.
[52] AiiDAlab Ansible role, URL: https://github.com/marvel-nccr/ansible-role-aiidalab.
[53] AiiDAlab-server Ansible role, URL: https://github.com/aiidalab/ansible-role-aiidalab-server.
[54] AiiDAlab deployment on Kubernetes, URL: https://github.com/aiidalab/aiidalab-k8s.
[55] A. Gourdon, On-Surface Synthesis, Advances in Atom and Single Molecule Machines, Springer International Publishing, Cham, 2016.
[56] J. Cai, P. Ruffieux, R. Jaafar, M. Bieri, T. Braun, S. Blankenburg, M. Muoth, A. P. Seitsonen, M. Saleh, X. Feng, K. Müllen, R. Fasel, Atomically precise bottom-up fabrication of graphene nanoribbons, Nature 466 (7305) (2010) 470–473, https://doi.org/10.1038/nature09211.
[57] L. Talirz, P. Ruffieux, R. Fasel, On-Surface Synthesis of Atomically Precise Graphene Nanoribbons, Adv. Mater. 28 (29) (2016) 6222–6231, https://doi.org/10.1002/adma.201505738.
[58] Empa nanoribbons app, URL: https://github.com/nanotech-empa/aiidalab-empa-nanoribbons.
[59] P. Giannozzi, S. Baroni, N. Bonini, M. Calandra, R. Car, C. Cavazzoni, D. Ceresoli, G.L. Chiarotti, M. Cococcioni, I. Dabo, A.D. Corso, S.d. Gironcoli, S. Fabris, G. Fratesi, R. Gebauer, U. Gerstmann, C. Gougoussis, A. Kokalj, M. Lazzeri, L. Martin-Samos, N. Marzari, F. Mauri, R. Mazzarello, S. Paolini, A. Pasquarello, L. Paulatto, C. Sbraccia, S. Scandolo, G. Sclauzero, A.P. Seitsonen, A. Smogunov, P. Umari, R.M. Wentzcovitch, Journal of Physics: Condensed Matter 21 (39) (2009) 395502, https://doi.org/10.1088/0953-8984/21/39/395502.
[60] Empa surfaces app, URL: https://github.com/nanotech-empa/aiidalab-empa-surfaces.
[61] J. Hutter, M. Iannuzzi, F. Schiffmann, J. VandeVondele, cp2k: atomistic simulations of condensed matter systems, Wiley Interdisciplinary Reviews: Computational Molecular, Science 4 (1) (2014) 15–25, https://doi.org/10.1002/wcms.1159.
[62] G. Mills, H. Jónsson, Quantum and thermal effects in H2 dissociative adsorption: Evaluation of free energy barriers in multidimensional quantum systems, Phys. Rev. Lett. 72 (7) (1994) 1124–1127, https://doi.org/10.1103/PhysRevLett.72.1124.
[63] G. Henkelman, B.P. Uberuaga, H. Jónsson, A climbing image nudged elastic band method for finding saddle points and minimum energy paths, J. Chem. Phys. 113 (22) (2000) 9901–9904, https://doi.org/10.1063/1.1329672.
[64] Empa scanning probe microscopy app, URL: https://github.com/nanotech-empa/aiidalab-empa-scanning-probe.
[65] Matplotlib: Visualization with Python, URL: https://matplotlib.org/.
[66] bqplot: Plotting library for IPython/Jupyter notebooks, URL: https://github.com/bqplot/bqplot.
[67] nglview, URL: https://github.com/nglviewer/nglview.
[68] A.H. Larsen, J.J. Mortensen, J. Blomqvist, I.E. Castelli, R. Christensen, J. Marcin Dułak, M.N. Friis, B. Groves, C. Hammer, E.D. Hargus, P.C. Hermes, P.B. Jennings, J. Jensen, J.R. Kermode, E.L. Kitchin, J. Kubal Kolsbjerg, S. Kristen Kaasbjerg, J. B. Lysgaard, T. Maronsson, T. Maxson, L. Pastewka Olsen, C. Andrew Peterson, J. Rostgaard, O. Schiøtz, M. Schütt, K.S. Strange, Tejs Vegge Thygesen, L. Vilhelmsen, M. Walter, Z. Zeng, K.W. Jacobsen, The atomic simulation environment—-a Python library for working with atoms, J. Phys.: Condens. Matter 29 (27) (2017), https://doi.org/10.1088/1361-648X/aa680e, 273002.