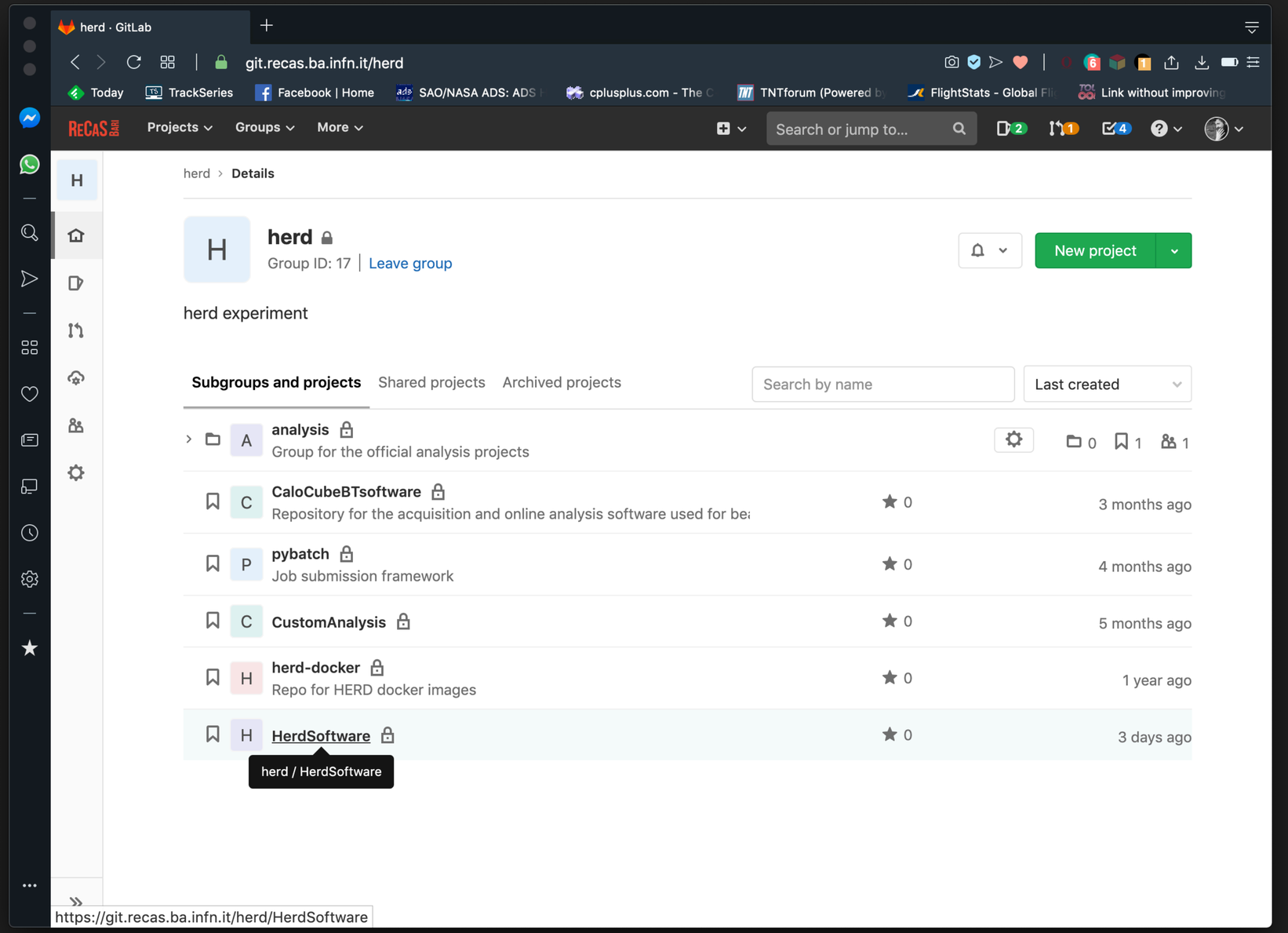
V. FORMATO - 17/03/2020

HERDSOFTWARE INFRASTRUCTURE

HERDSOFTWARE

HerdSoftware is currently hosted on a GitLab instance at RECAS

The code is organized under a "herd" group. Here we group all projects: starting from the **HerdSoftware** repo, up to analysis code by the users, or production tools for job submission.



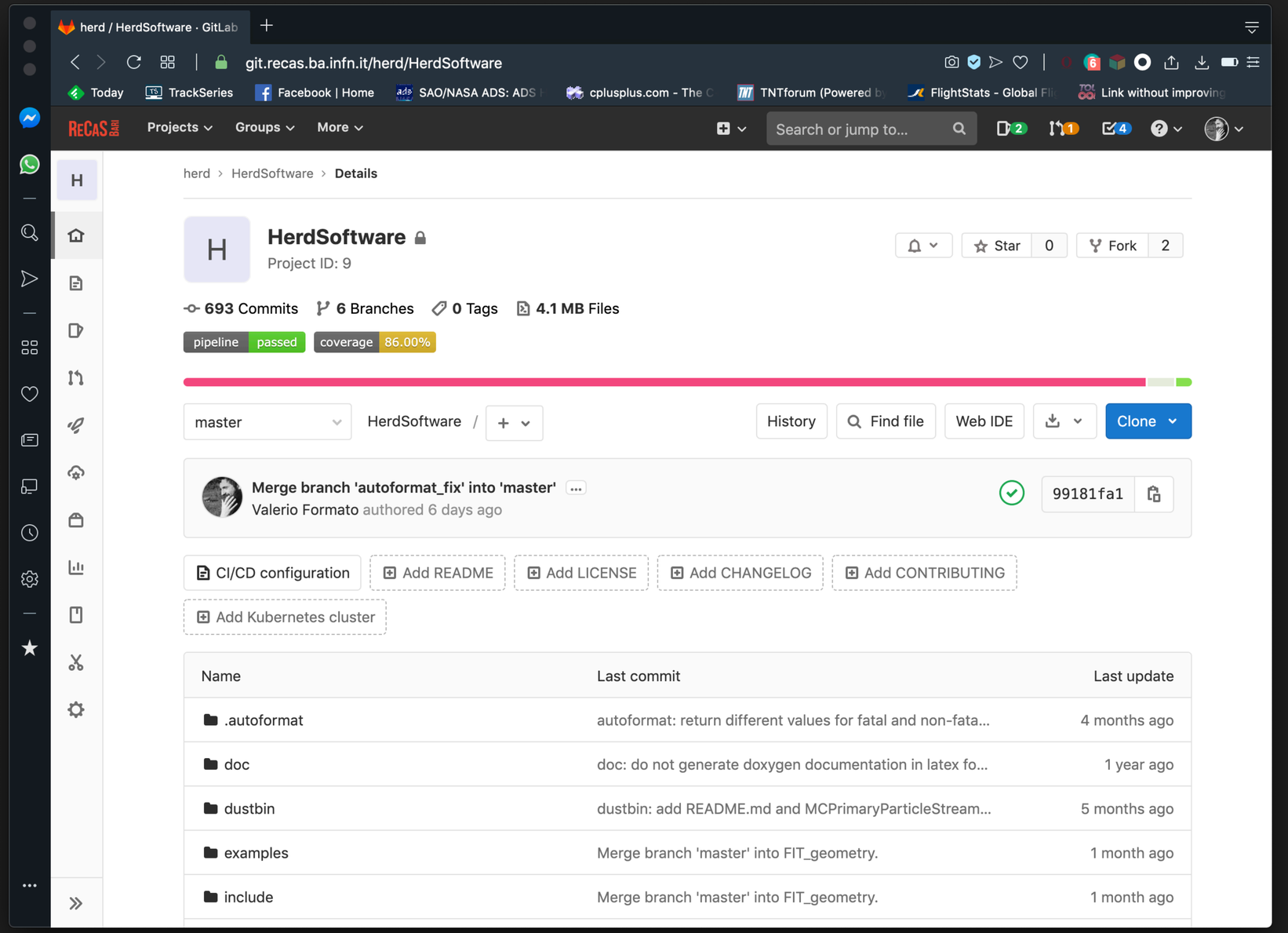
HERDSOFTWARE

Every *git* repository in *gitlab* is associated to a **project** (https://docs.gitlab.com/ee/user/project/).

Projects encompass a *git* repo and offer additional tools around it:

- Issue tracker
- Merge requests
- Continuous integration
- Wiki documentation
- ...and many more

We will touch all these subject briefly

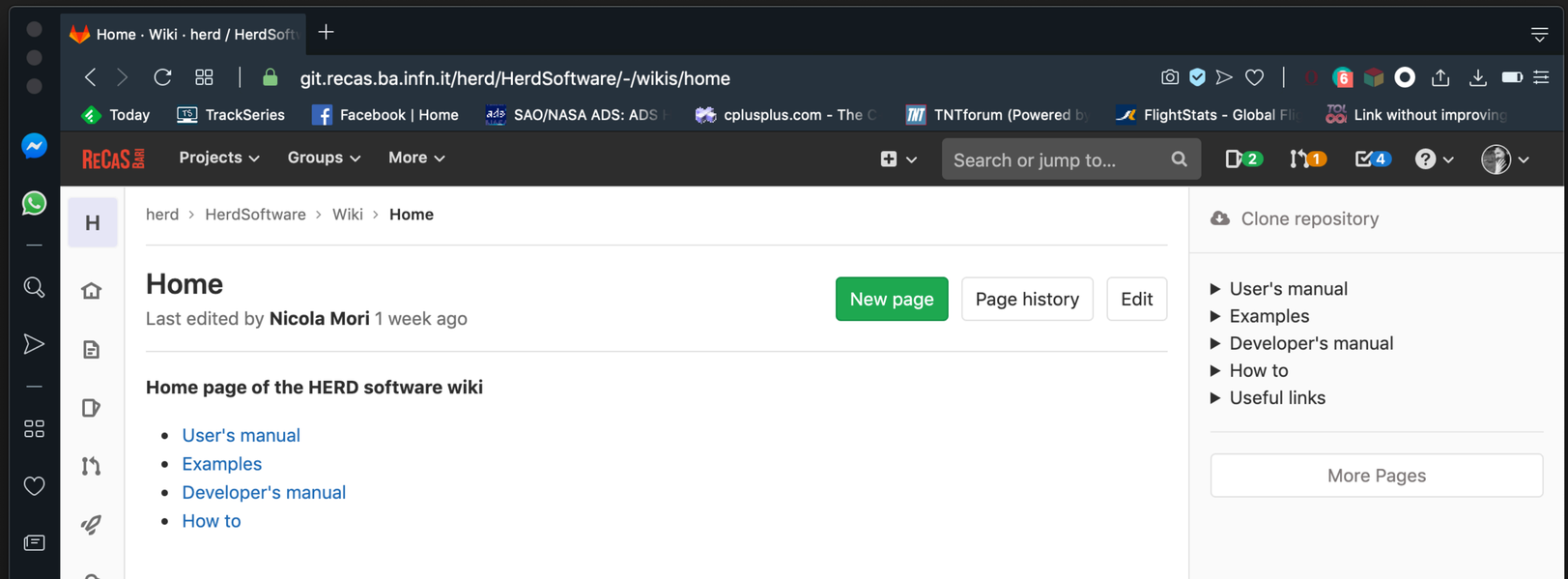


WIKI

A separate system for documentation called **Wiki**, is built right into each GitLab project. It is enabled by default on all new projects and you can find it under **Wiki** in your project.

Wikis are very convenient if you don't want to keep your documentation in your repository, but you do want to keep it in the same project where your code resides.

HerdSoftware comes with its own **Wiki** documentation that we try to keep always updated. You are encouraged to consult it whenever you have doubts about the software. If you think that some section is not described in enough detail, don't hesitate to write a mail or open an issue (see later...)

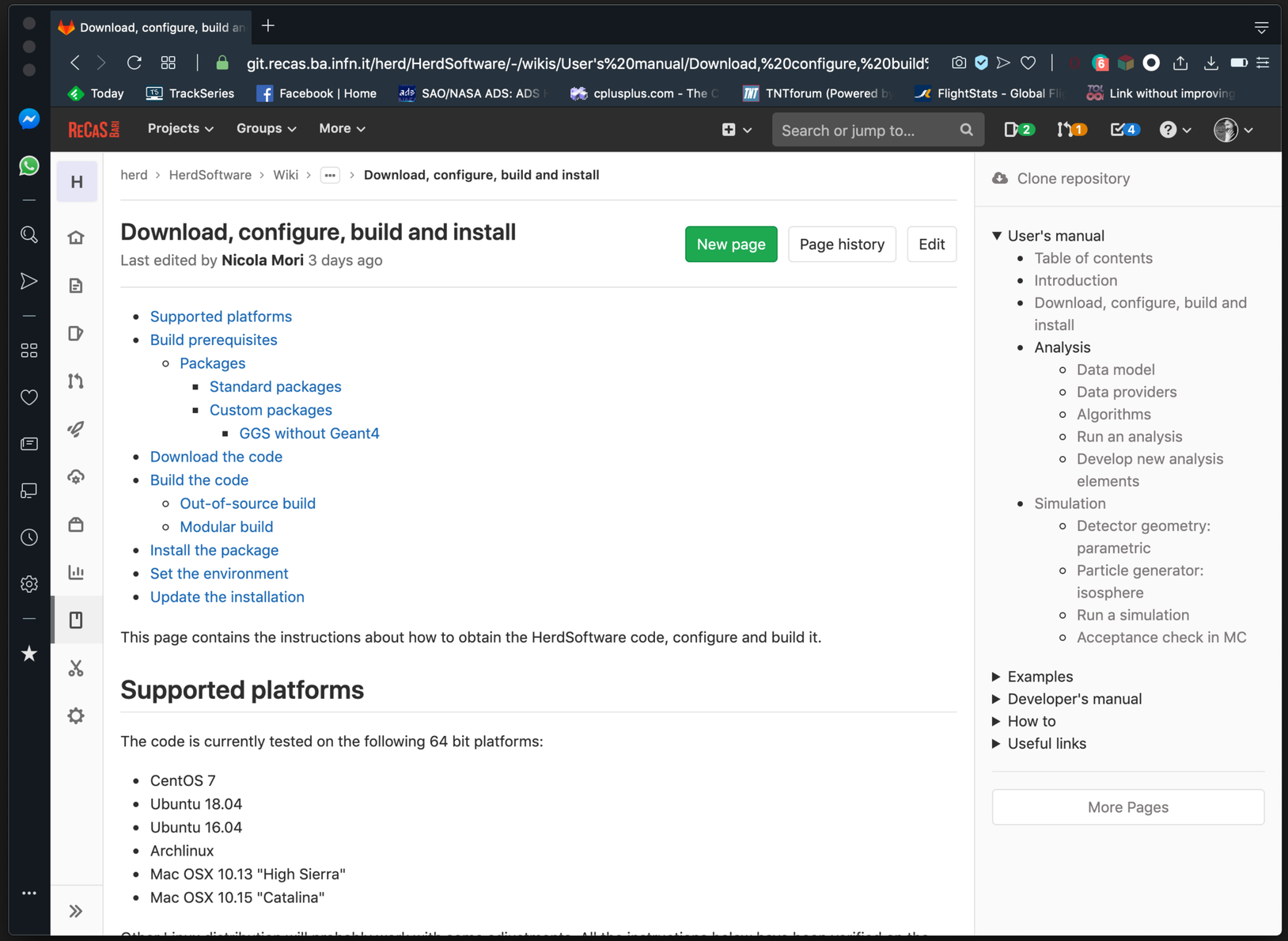


WIKI

We try to keep the HerdSoftware wiki as updated and complete as possible. This is supposed to be the entry point for every user needing help or information about the data format / algorithms / etc...

The User's manual tells you:

- How to install the project

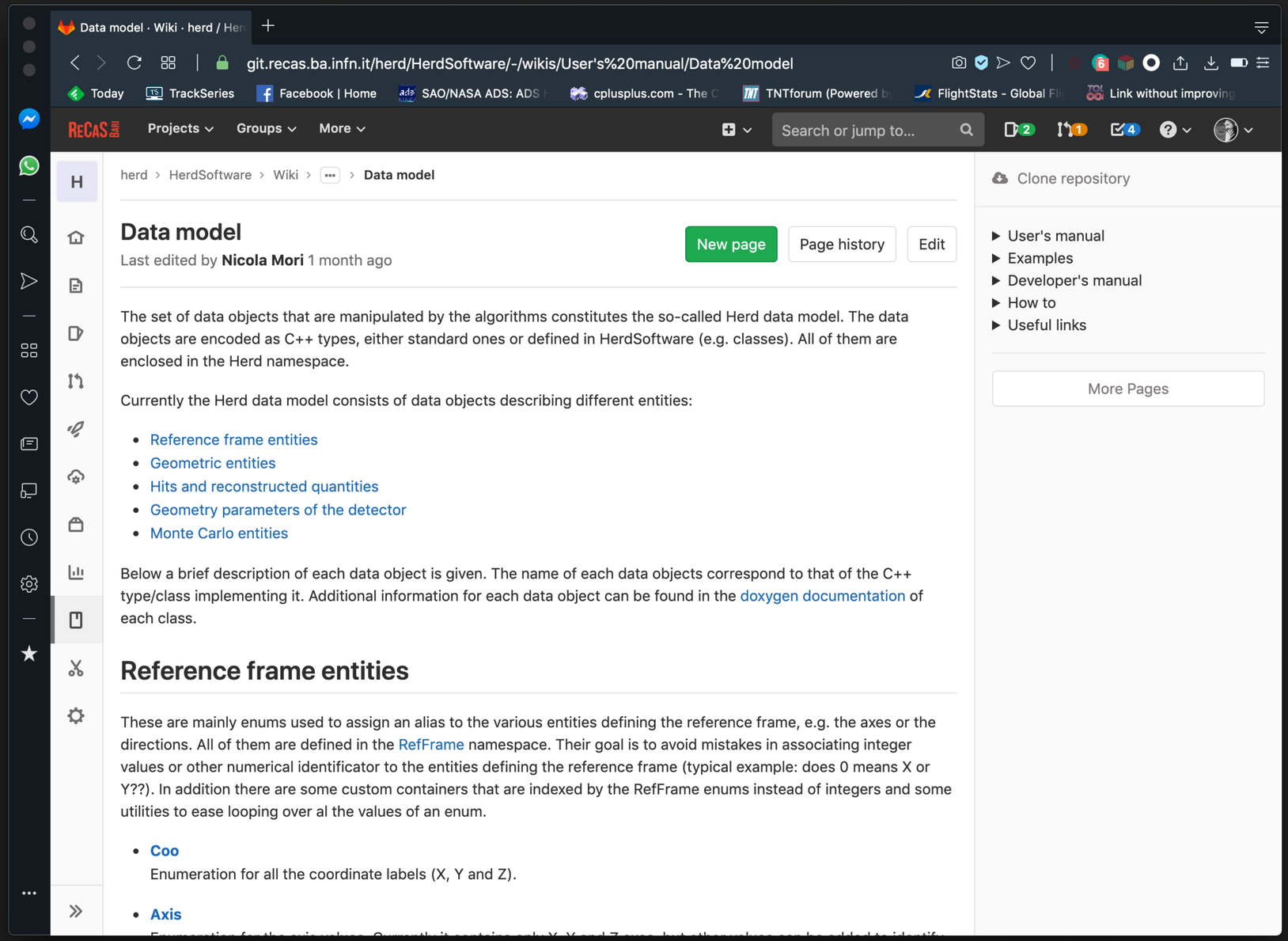


WIKI

We try to keep the HerdSoftware wiki as updated and complete as possible. This is supposed to be the entry point for every user needing help or information about the data format / algorithms / etc...

The User's manual tells you:

- How to install the project
- How the data model is organized

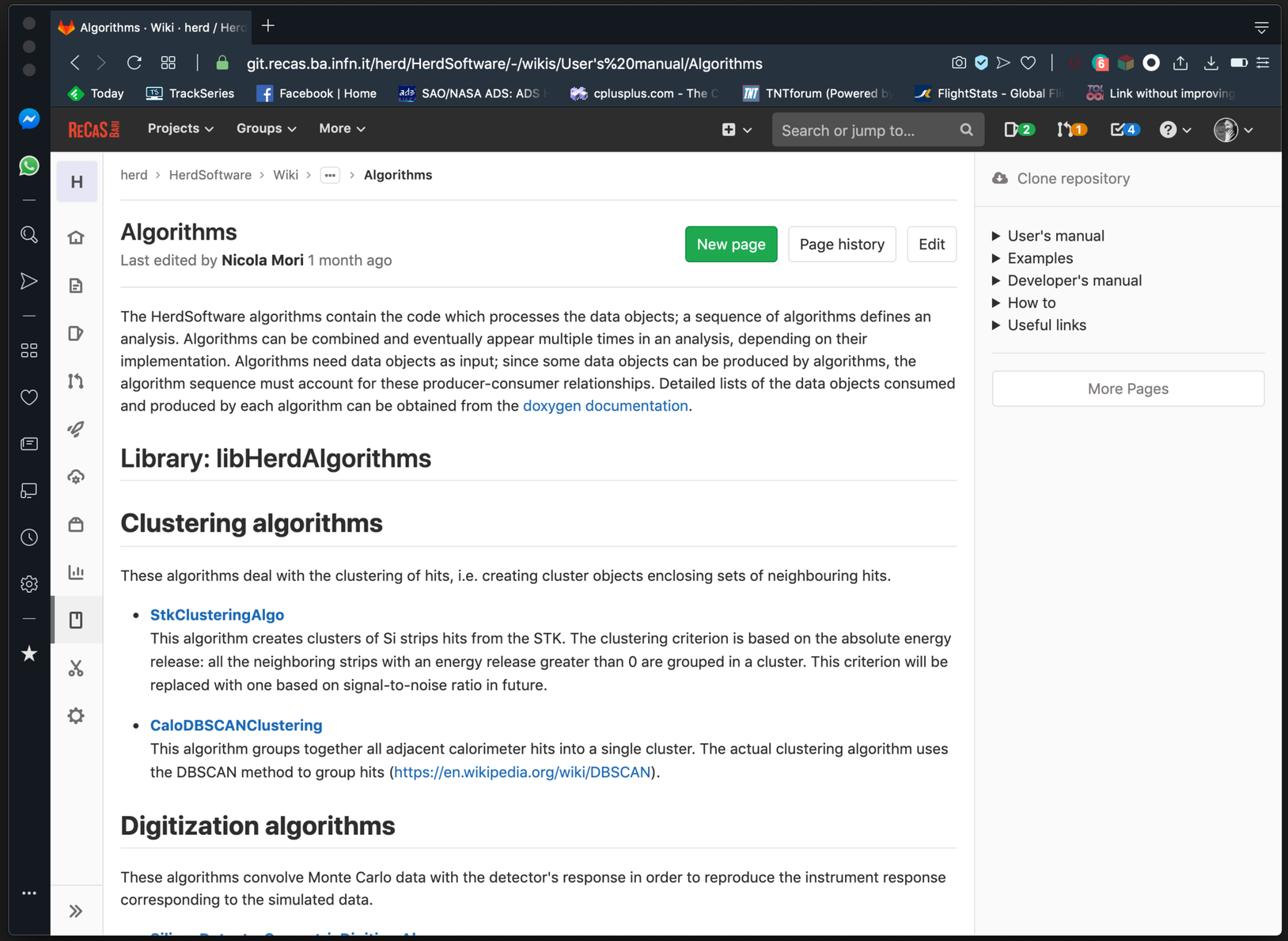


WIKI

We try to keep the HerdSoftware wiki as updated and complete as possible. This is supposed to be the entry point for every user needing help or information about the data format / algorithms / etc...

The User's manual tells you:

- How to install the project
- How the data model is organized
- What algorithms are available and how to use them

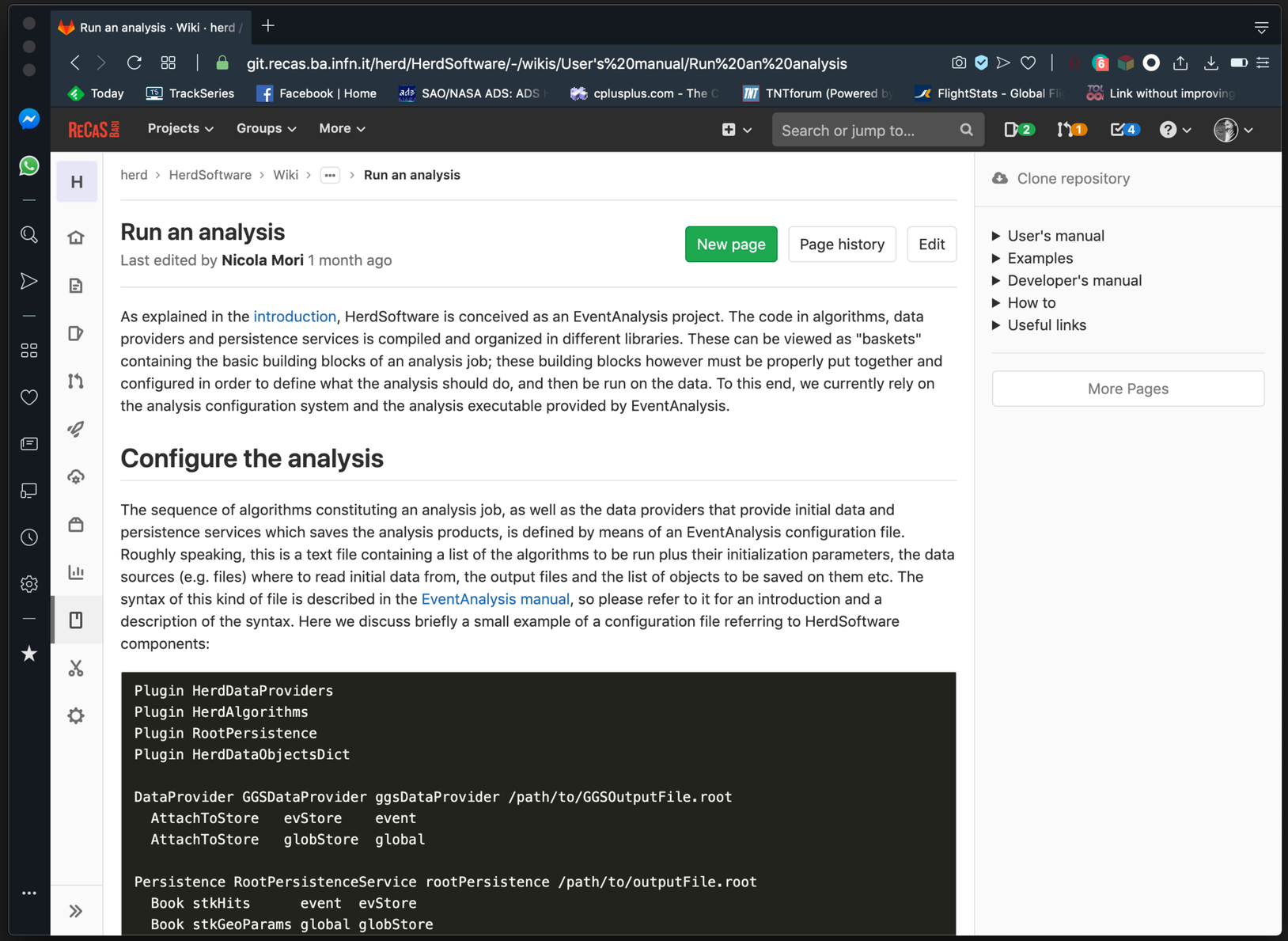


WIKI

We try to keep the HerdSoftware wiki as updated and complete as possible. This is supposed to be the entry point for every user needing help or information about the data format / algorithms / etc...

The User's manual tells you:

- How to install the project
- How the data model is organized
- What algorithms are available and how to use them
- How to run an analysis

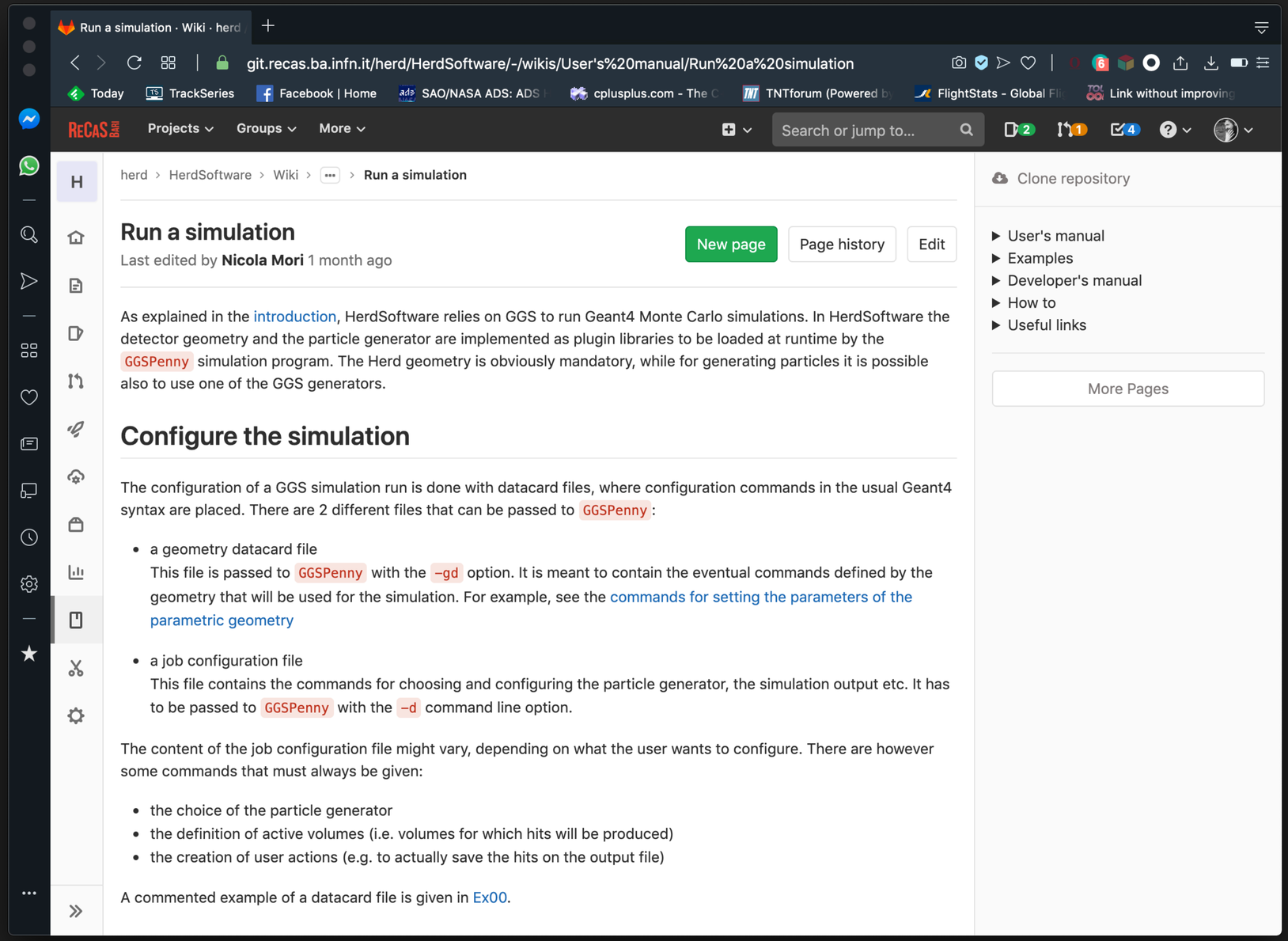


WIKI

We try to keep the HerdSoftware wiki as updated and complete as possible. This is supposed to be the entry point for every user needing help or information about the data format / algorithms / etc...

The User's manual tells you:

- How to install the project
- How the data model is organized
- What algorithms are available and how to use them
- How to run an analysis
- How to run a simulation

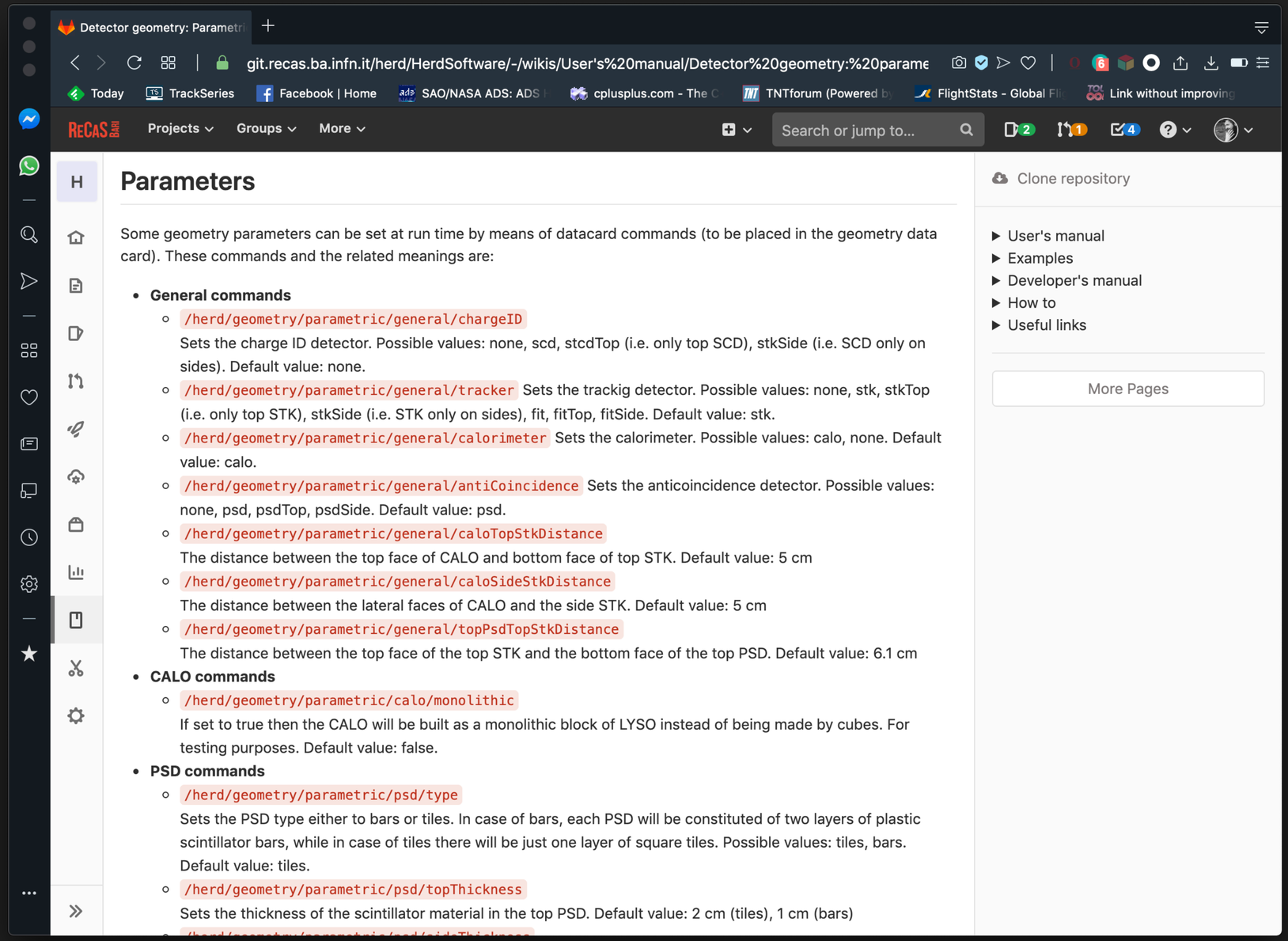


WIKI

We try to keep the HerdSoftware wiki as updated and complete as possible. This is supposed to be the entry point for every user needing help or information about the data format / algorithms / etc...

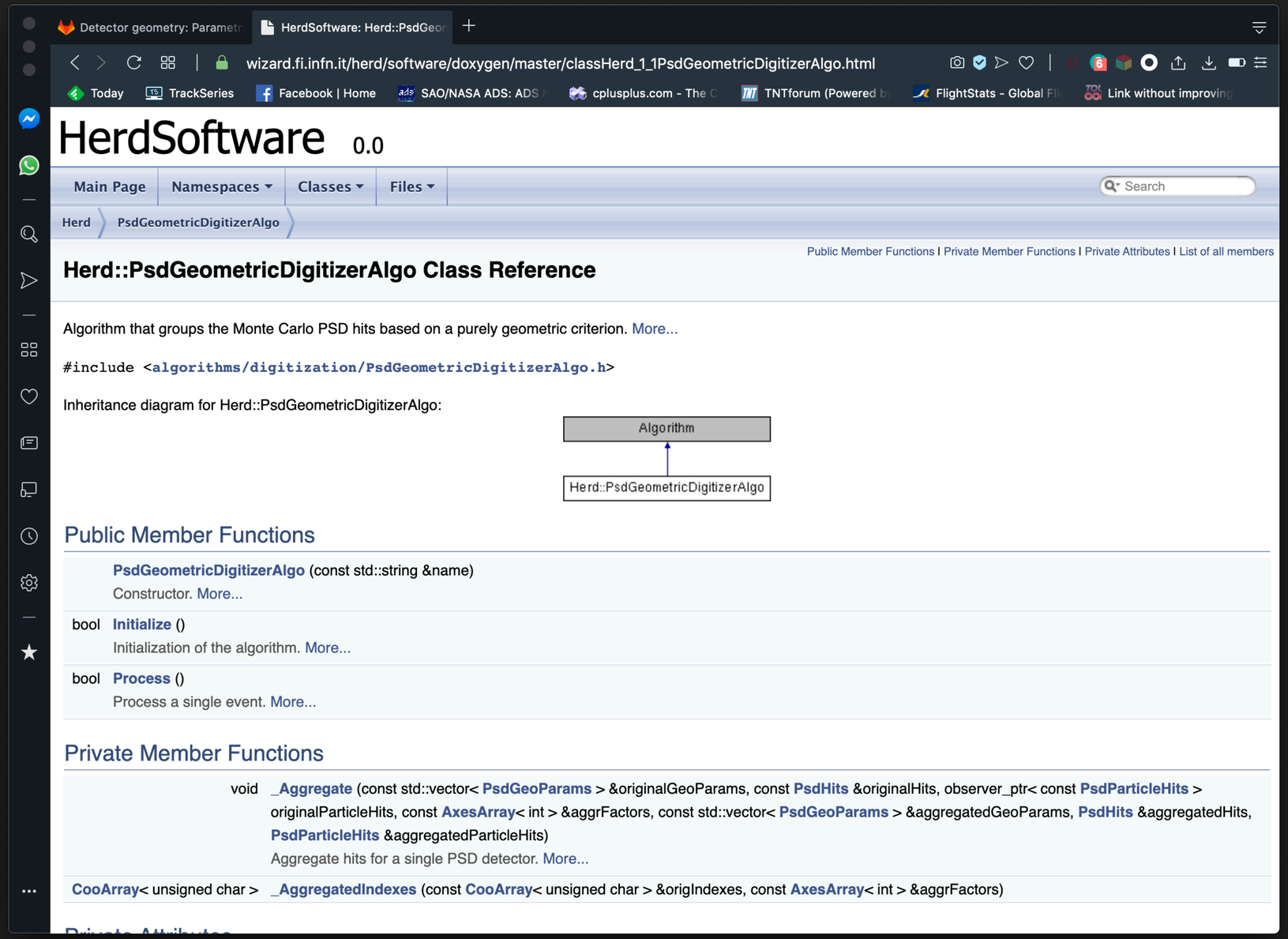
The User's manual tells you:

- How to install the project
- How the data model is organized
- What algorithms are available and how to use them
- How to run an analysis
- How to run a simulation
- How to customize the detector geometry
- ...



DOXYGEN

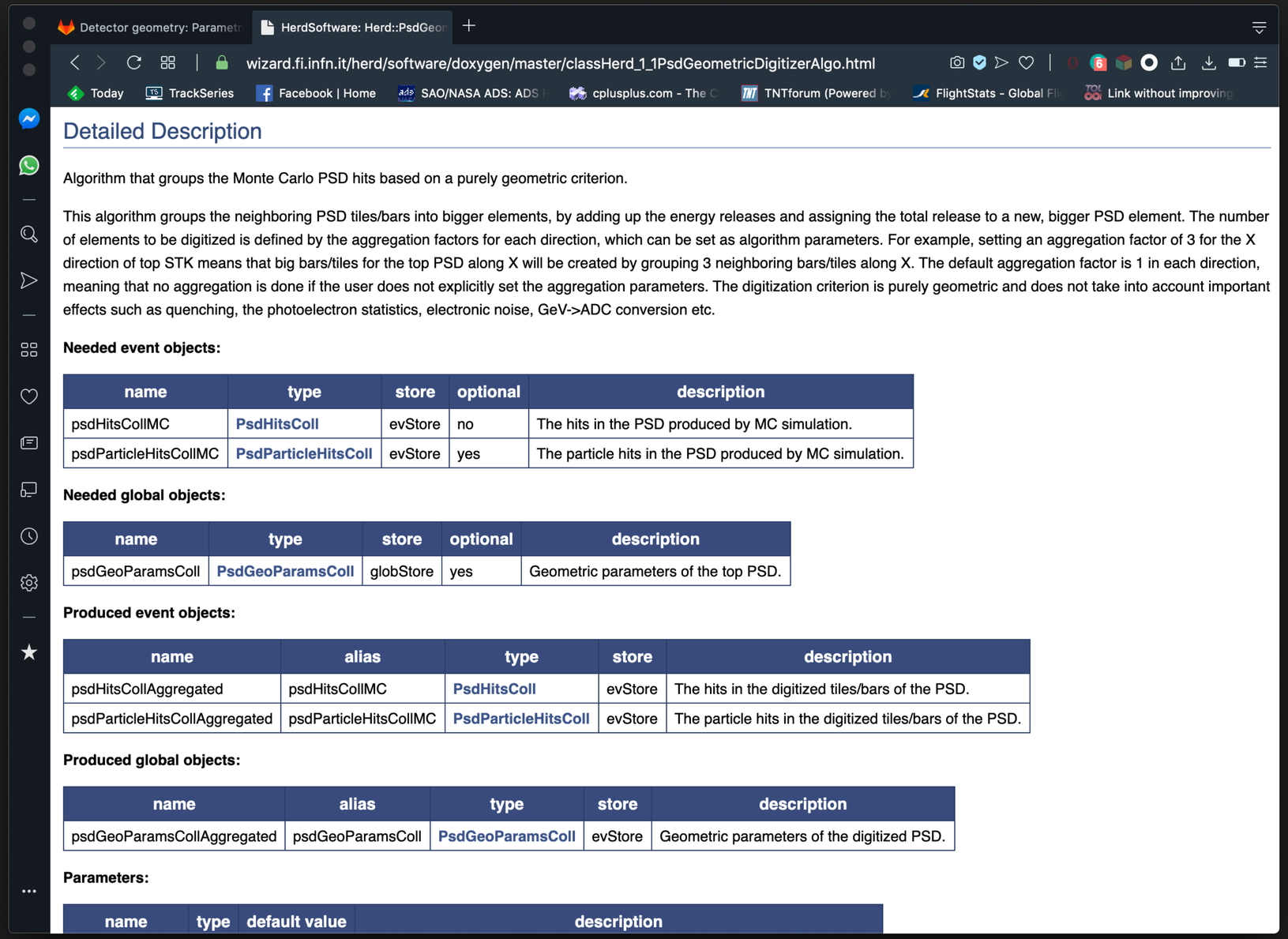
Complementing the Wiki we also have a Doxygen documentation with a detailed description of the implementation for all the relevant classes.



DOXYGEN

Complementing the Wiki we also have a Doxygen documentation with a detailed description of the implementation for all the relevant classes.

Algorithms also have a full description of all the data objects they need and/or produce.



Detailed Description

Algorithm that groups the Monte Carlo PSD hits based on a purely geometric criterion.

This algorithm groups the neighboring PSD tiles/bars into bigger elements, by adding up the energy releases and assigning the total release to a new, bigger PSD element. The number of elements to be digitized is defined by the aggregation factors for each direction, which can be set as algorithm parameters. For example, setting an aggregation factor of 3 for the X direction of top STK means that big bars/tiles for the top PSD along X will be created by grouping 3 neighboring bars/tiles along X. The default aggregation factor is 1 in each direction, meaning that no aggregation is done if the user does not explicitly set the aggregation parameters. The digitization criterion is purely geometric and does not take into account important effects such as quenching, the photoelectron statistics, electronic noise, GeV->ADC conversion etc.

Needed event objects:

name	type	store	optional	description
psdHitsCollMC	PsdHitsColl	evStore	no	The hits in the PSD produced by MC simulation.
psdParticleHitsCollMC	PsdParticleHitsColl	evStore	yes	The particle hits in the PSD produced by MC simulation.

Needed global objects:

name	type	store	optional	description
psdGeoParamsColl	PsdGeoParamsColl	globStore	yes	Geometric parameters of the top PSD.

Produced event objects:

name	alias	type	store	description
psdHitsCollAggregated	psdHitsCollMC	PsdHitsColl	evStore	The hits in the digitized tiles/bars of the PSD.
psdParticleHitsCollAggregated	psdParticleHitsCollMC	PsdParticleHitsColl	evStore	The particle hits in the digitized tiles/bars of the PSD.

Produced global objects:

name	alias	type	store	description
psdGeoParamsCollAggregated	psdGeoParamsColl	PsdGeoParamsColl	evStore	Geometric parameters of the digitized PSD.

Parameters:

name	type	default value	description

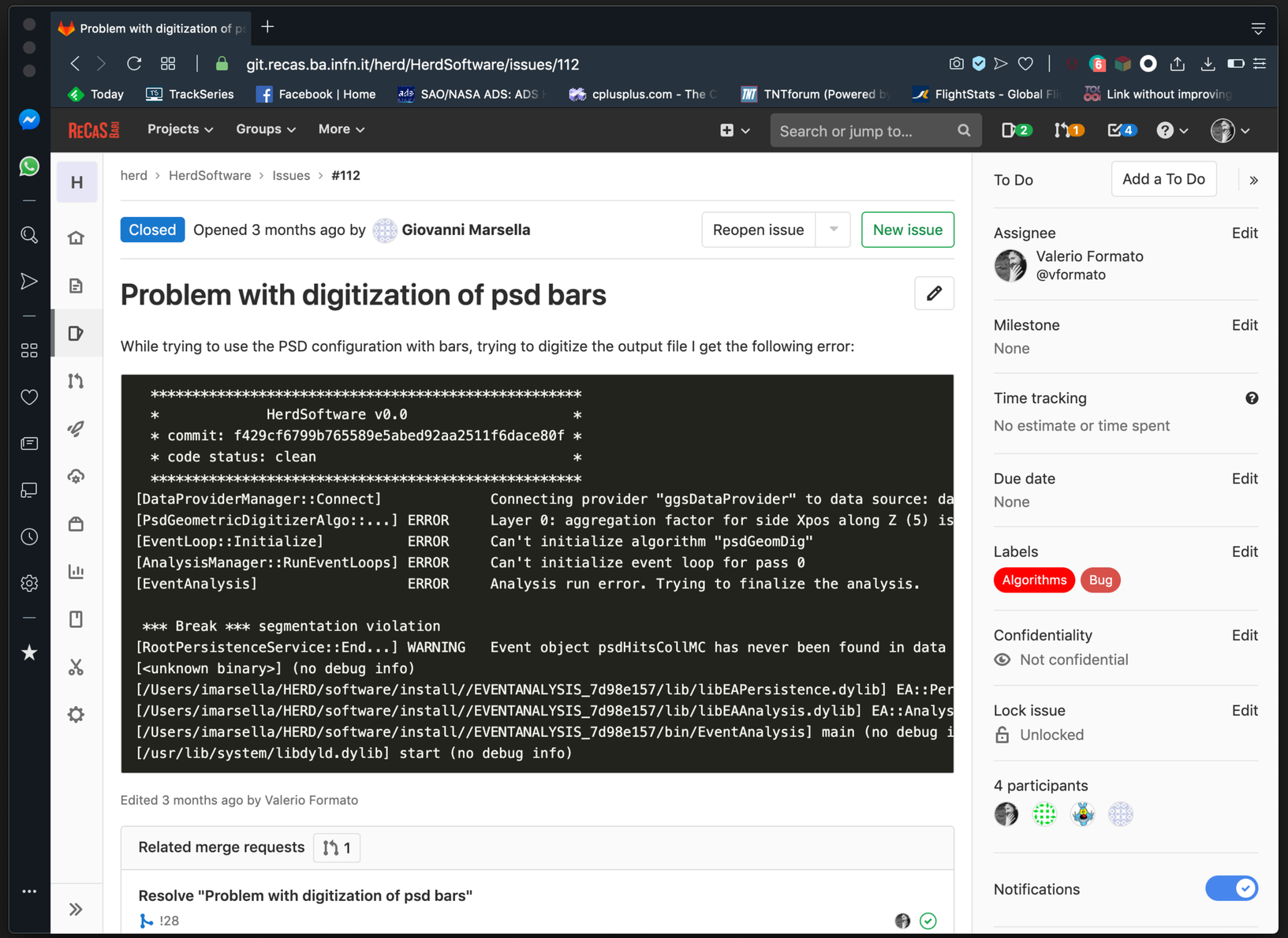
ISSUES

Issues are the fundamental medium for collaborating on ideas and planning work in GitLab.

In HerdSoftware they are used for:

- Report a bug
- Propose a new feature or work item
- Organise and split the work on big tasks between different developers

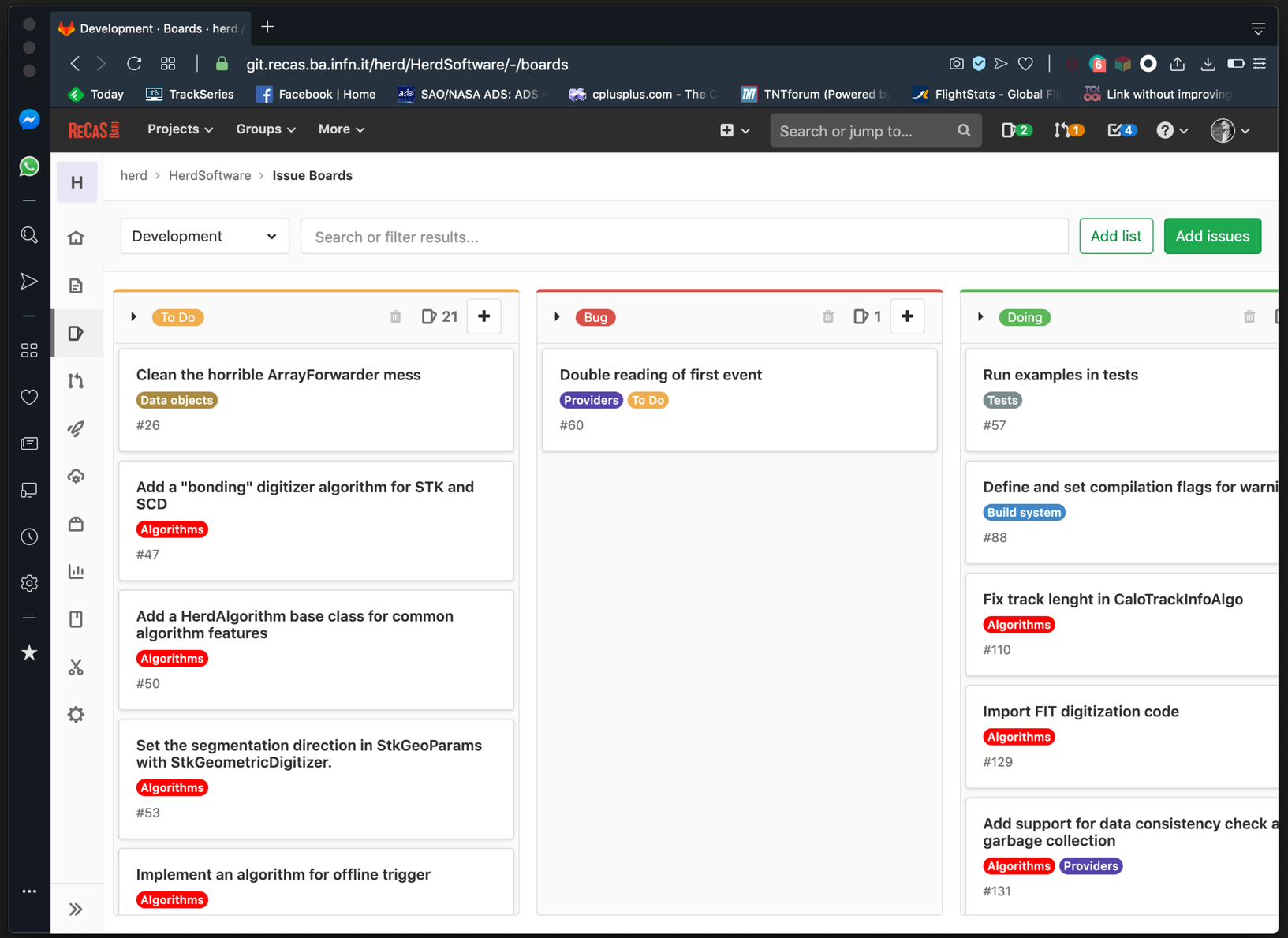
Each issue comes with a title and a description. Comments can be added as the work progresses. Gitlab can automatically create a dedicated branch (and a merge request) for a given issue. Issues can be labelled with custom with tags.



ISSUES

In HerdSoftware they are used for:

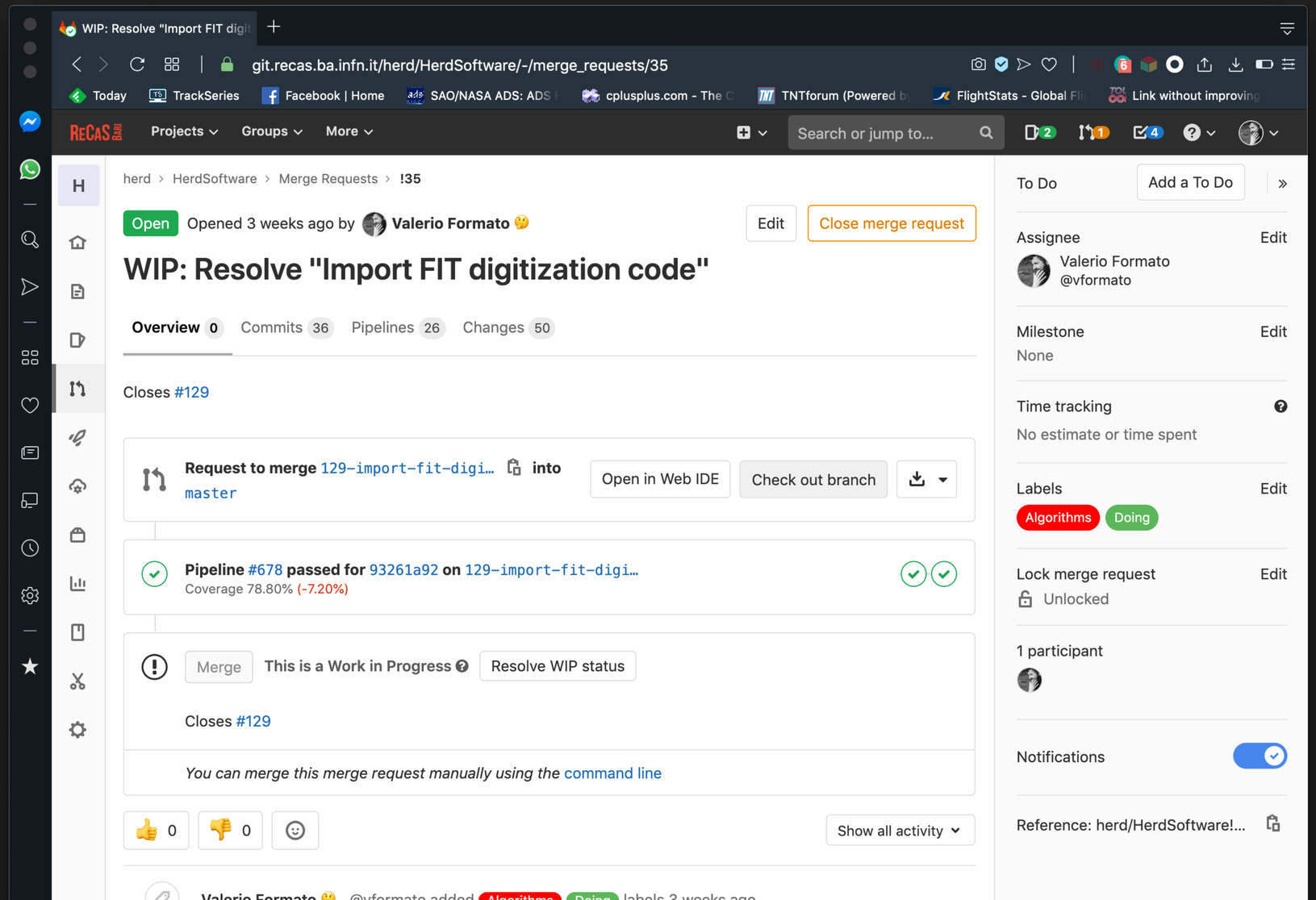
- Report a bug
- Propose a new feature or work item
- Organise and split the work on big tasks between different developers



ADVANCED TOOLS: MERGE REQUESTS

A Merge Request (MR) is the basis of GitLab as a code collaboration and version control platform. It is as simple as the name implies: a request to merge one branch into another.

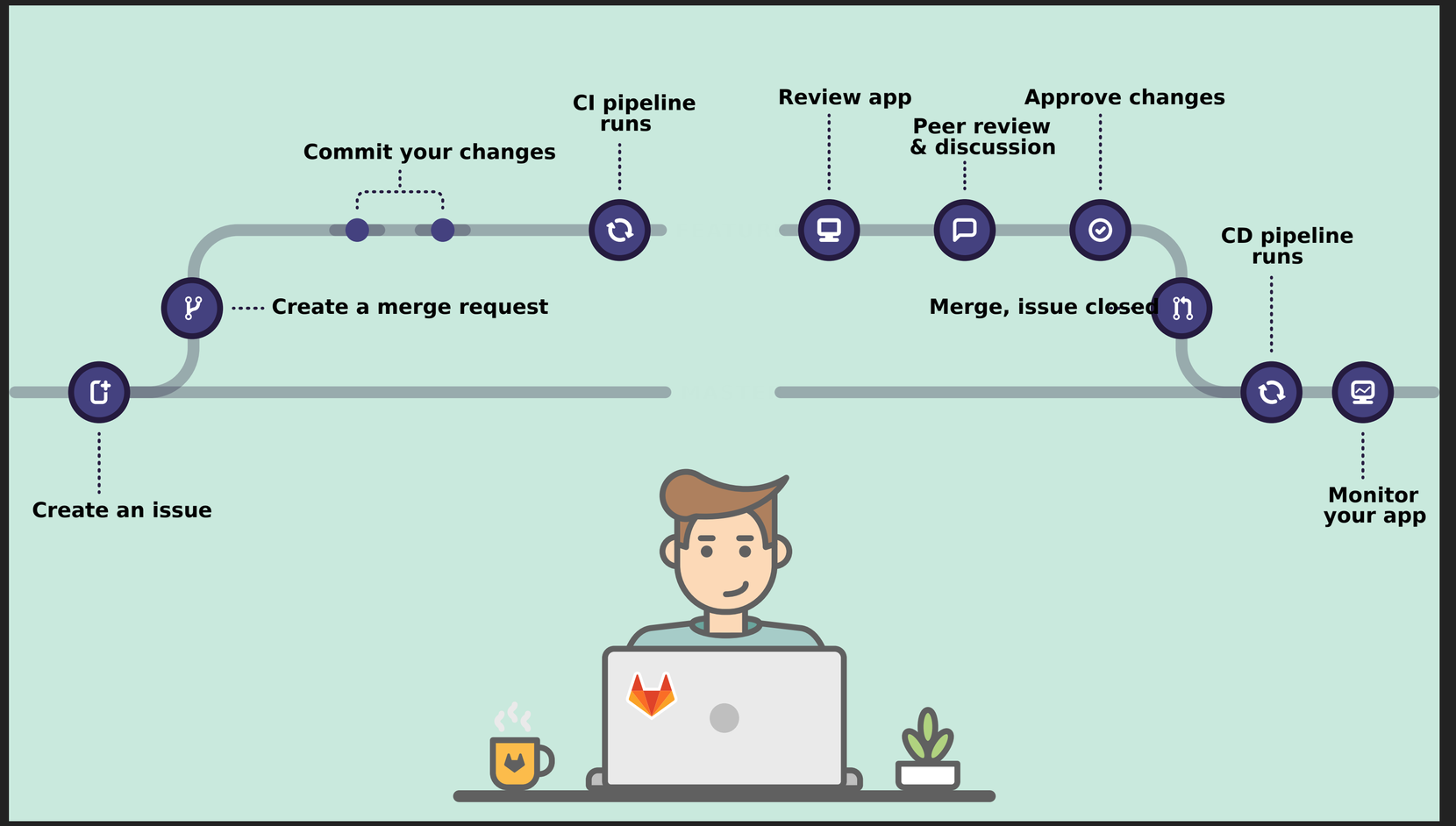
- Compare the changes between two branches
- Review and discuss the proposed modifications inline
- Build, test, and deploy your code in a per-branch basis with built-in GitLab CI/CD
- Automatically close the issue(s) that originated the implementation proposed in the merge request
- Organize your issues and merge requests consistently throughout the project with labels
- Resolve merge conflicts from the UI
- Enable fast-forward merge requests



ADVANCED TOOLS: MERGE REQUESTS

A Merge Request (MR) is the basis of GitLab as a code collaboration and version control platform. It is as simple as the name implies: a request to merge one branch into another.

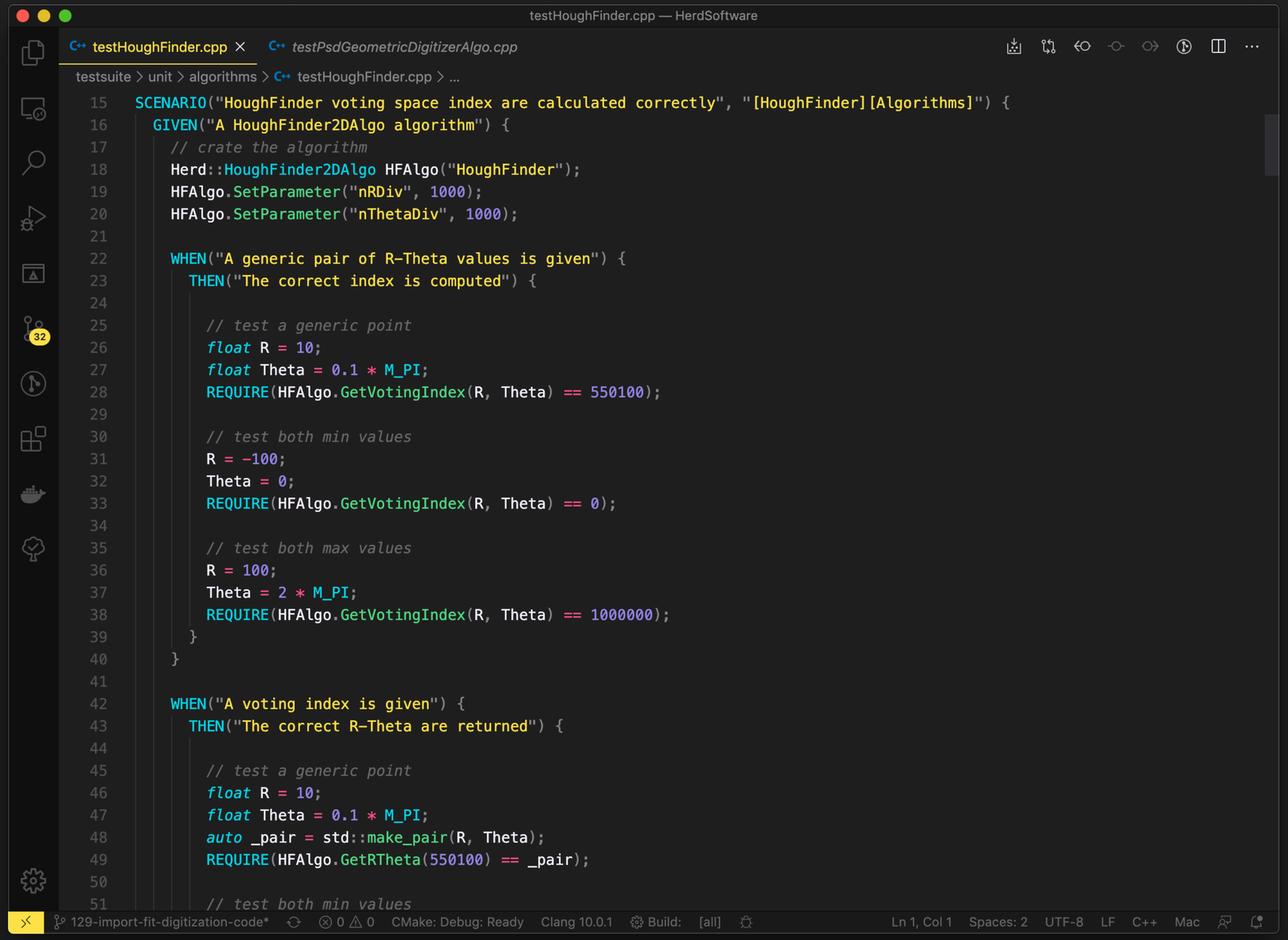
- Compare the changes between two branches
- Review and discuss the proposed modifications inline
- Build, test, and deploy your code in a per-branch basis with built-in GitLab CI/CD
- Automatically close the issue(s) that originated the implementation proposed in the merge request
- Organize your issues and merge requests consistently throughout the project with labels
- Resolve merge conflicts from the UI
- Enable fast-forward merge requests



UNIT-TESTS

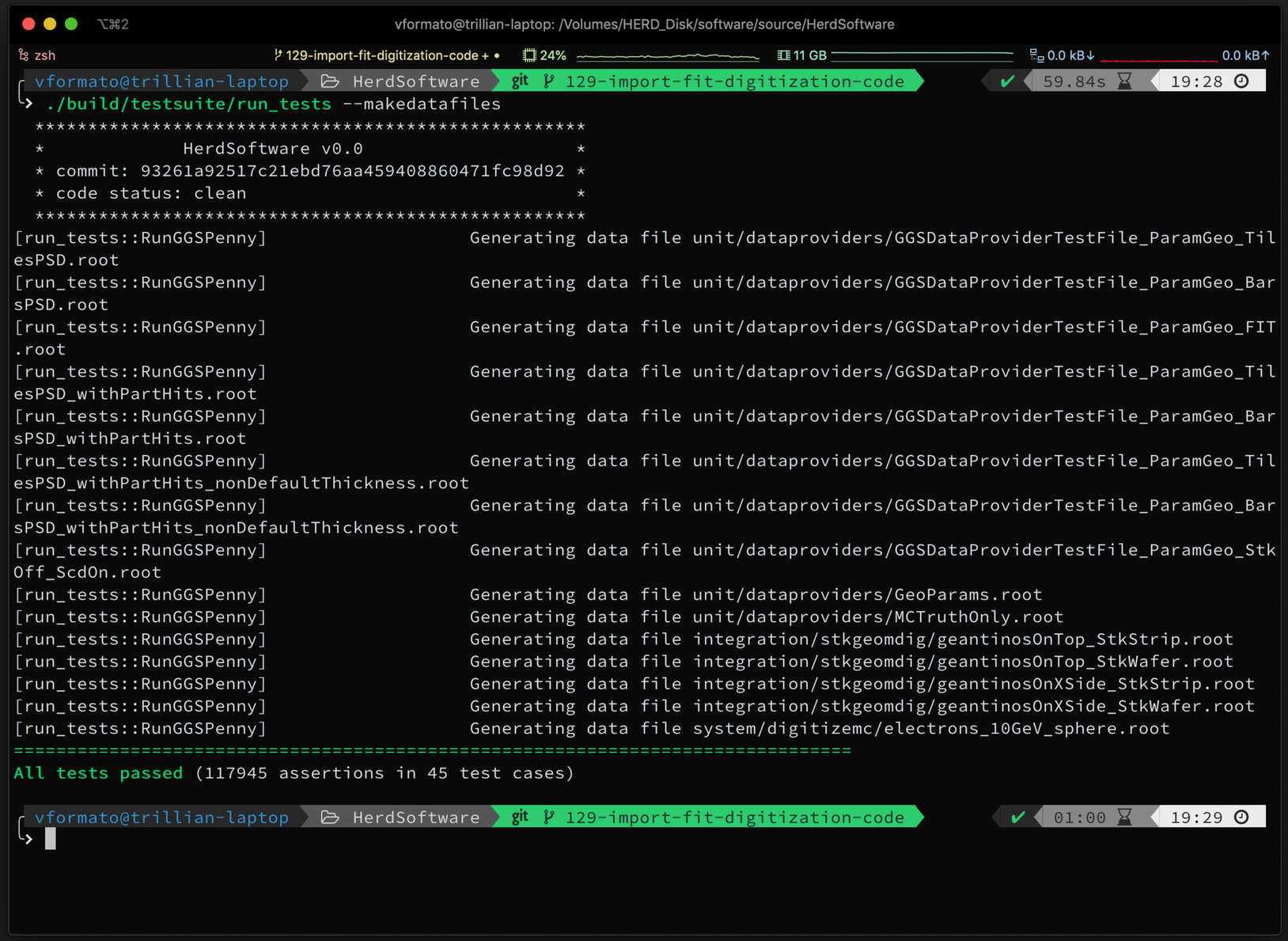
HerdSoftware comes with a suite of tests, based on the Catch2 framework, divided into several categories:

- Unit tests:
  Make sure that the each single component of HerdSoftware perform its function as expected.
- Integration tests:
  Check that details implemented in the simulation correspond to their corresponding software-side implementation
- System tests:
  Check the whole pipeline for producing and analysing data
- External projects tests:
  Check that the API for user-implemented algorithms is stable.



UNIT-TESTS

This is extremely important: having a test suite while developing a software allows the developers to spot any breaking changes in the code as soon as possible, so that they will know immediately if any new feature they are working on breaks any component and they can find and fix any newly introduced bug before pushing their work.

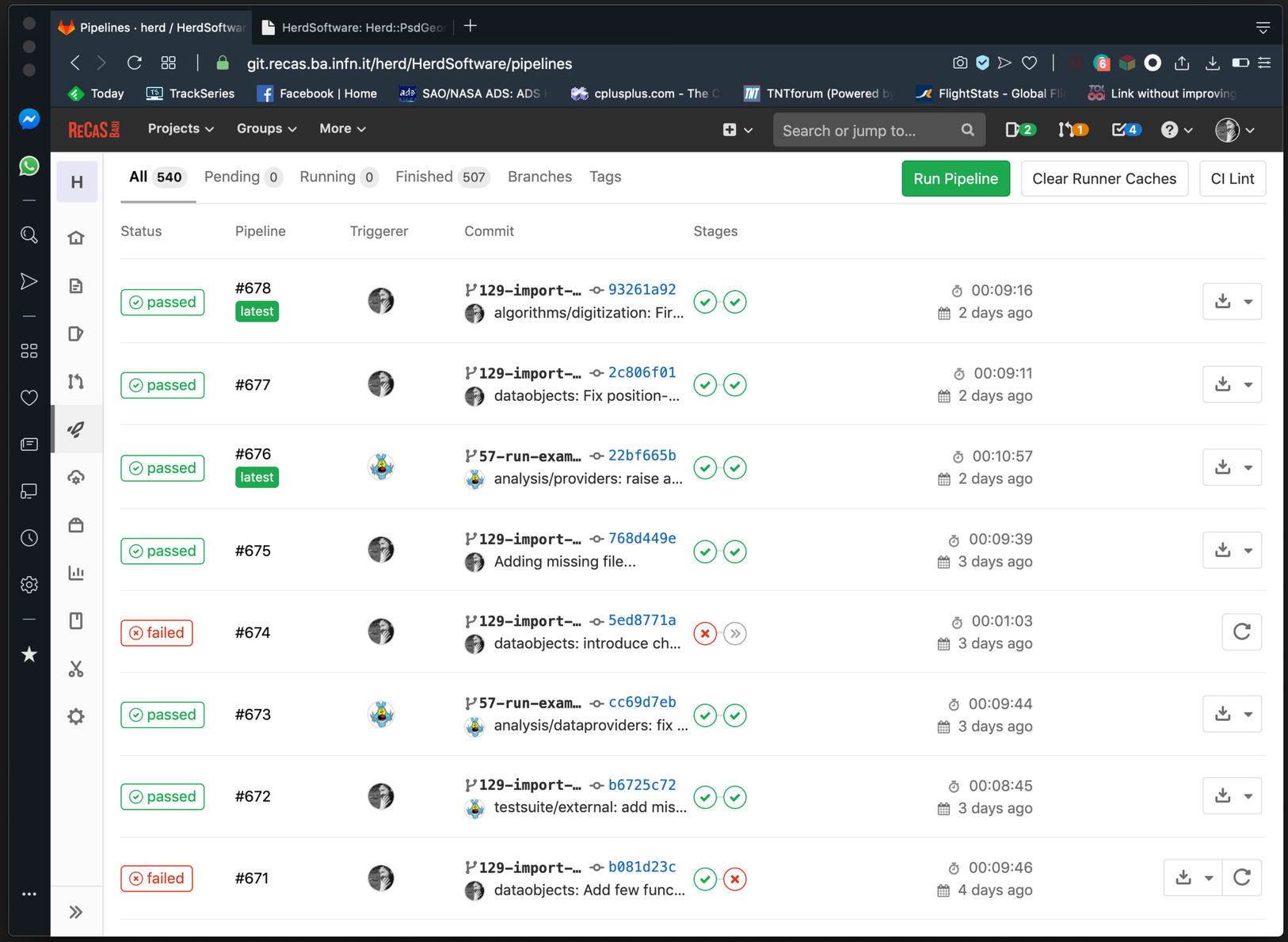


CONTINUOUS INTEGRATION

What if the developer forgets to run the test while he is working?

Or if he doesn't try to compile the code before pushing it?

We have configured HerdSoftware to use the GitLab continuous integration facility to compile the code and run the tests automatically on each push. If anything breaks the developer receives an automatically generated email with detailed information on what's gone wrong.



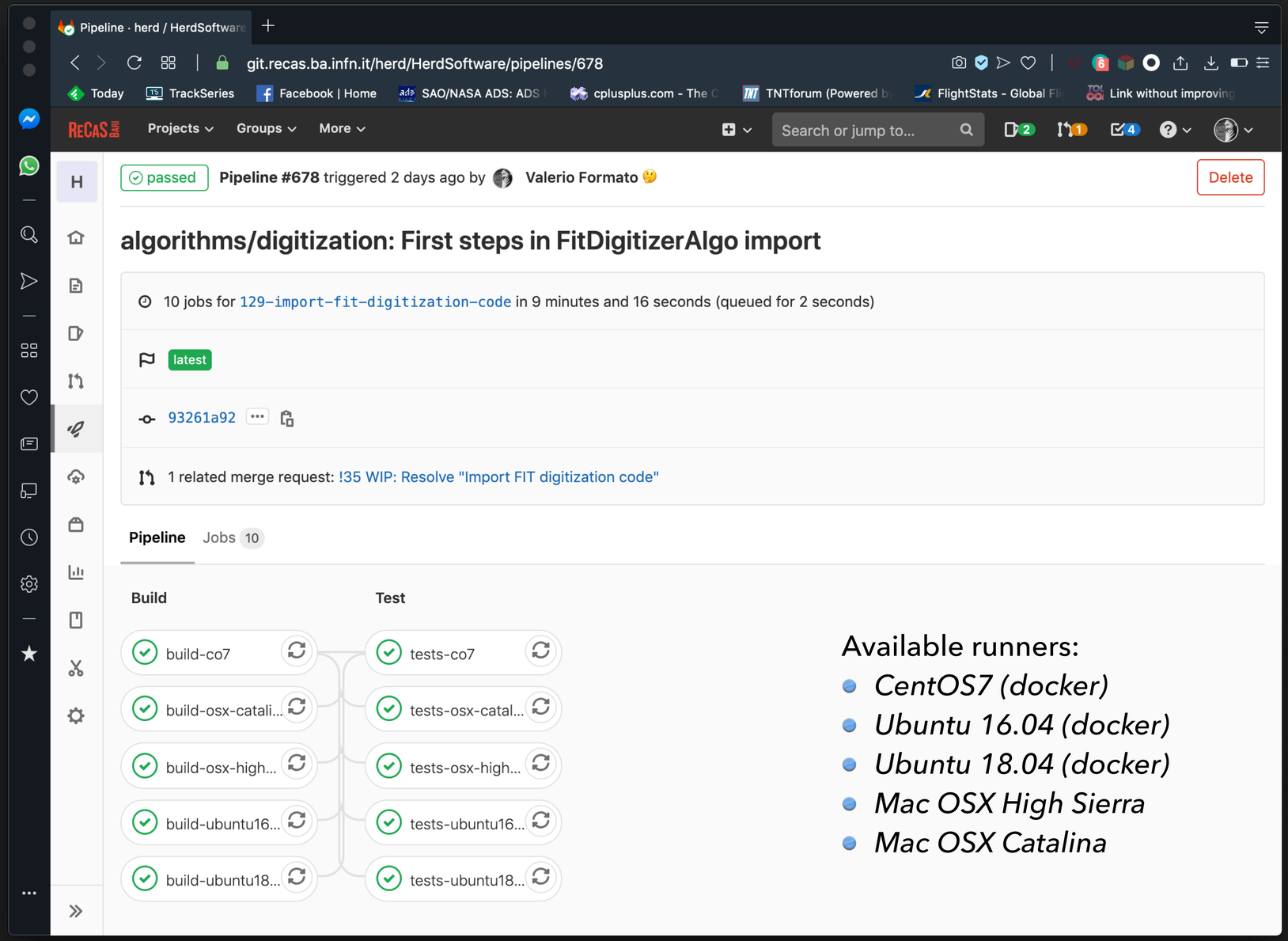
CONTINUOUS INTEGRATION

What if the developer forgets to run the test while he is working?

Or if he doesn't try to compile the code before pushing it?

We have configured HerdSoftware to use the GitLab continuous integration facility to compile the code and run the tests automatically on each push. If anything breaks the developer receives an automatically generated email with detailed information on what's gone wrong.

Moreover the code is built and tested on several platforms in order to make sure that it works properly on all the supported platforms.



Available runners:

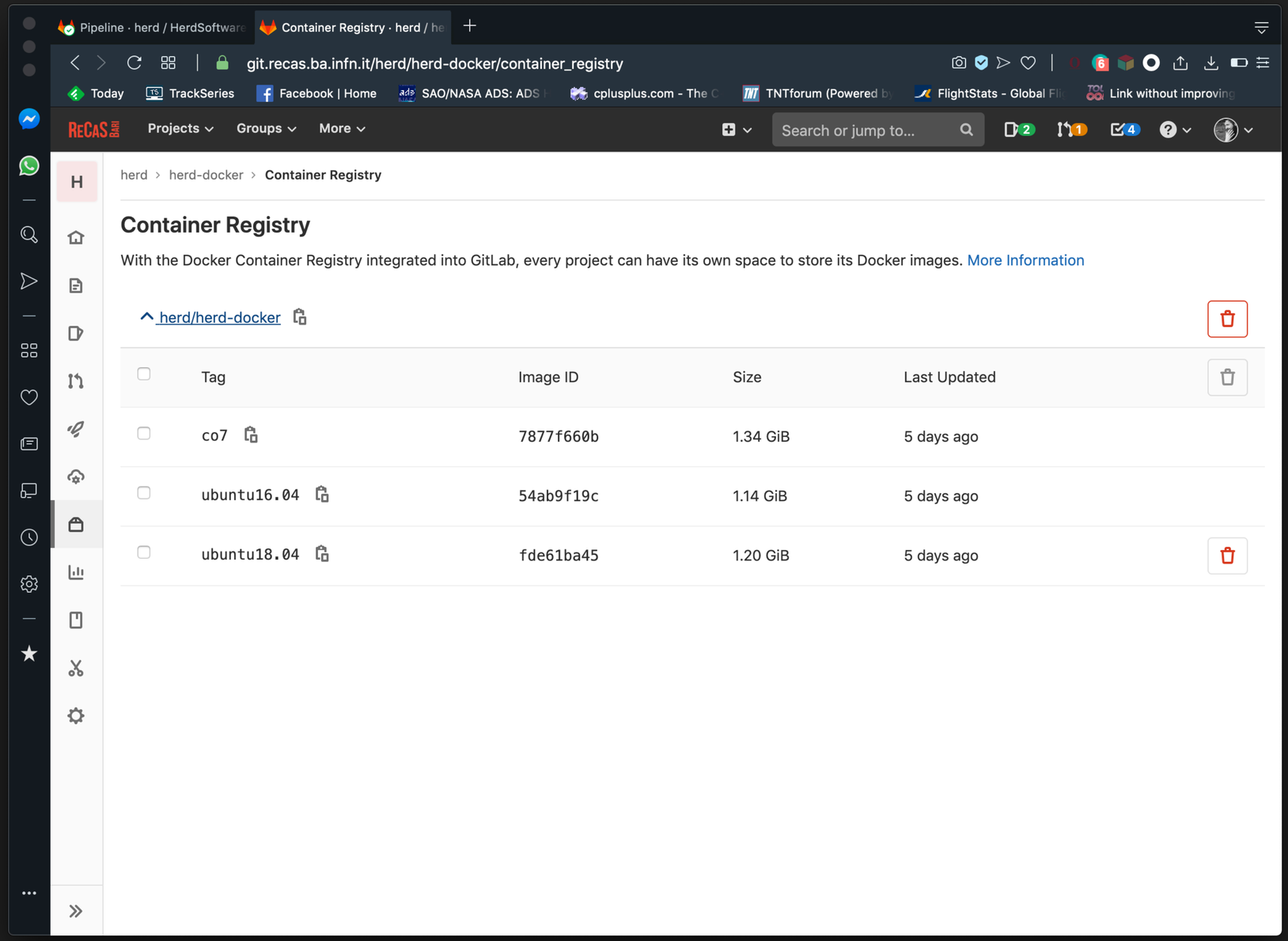
- *CentOS7 (docker)*
- *Ubuntu 16.04 (docker)*
- *Ubuntu 18.04 (docker)*
- *Mac OSX High Sierra*
- *Mac OSX Catalina*

CONTINUOUS INTEGRATION

We have configured HerdSoftware to use the GitLab continuous integration facility to compile the code and run the tests automatically on each push. If anything breaks the developer receives an automatically generated email with detailed information on what's gone wrong.

Moreover the code is built and tested on several platforms in order to make sure that it works properly on all the supported platforms.

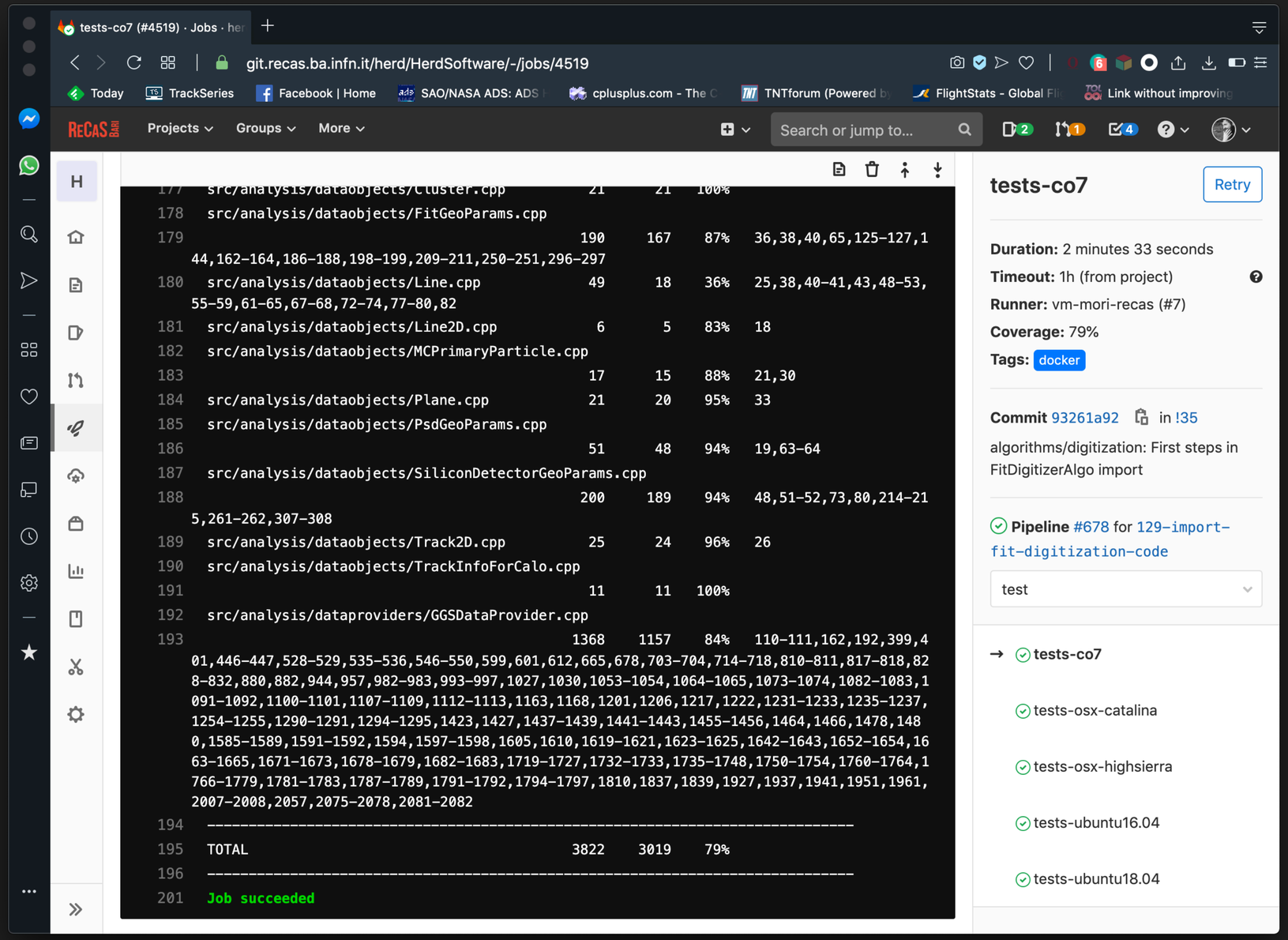
Linux distributions are tested on custom docker images automatically generated in another GitLab project and are available for debugging.



CONTINUOUS INTEGRATION

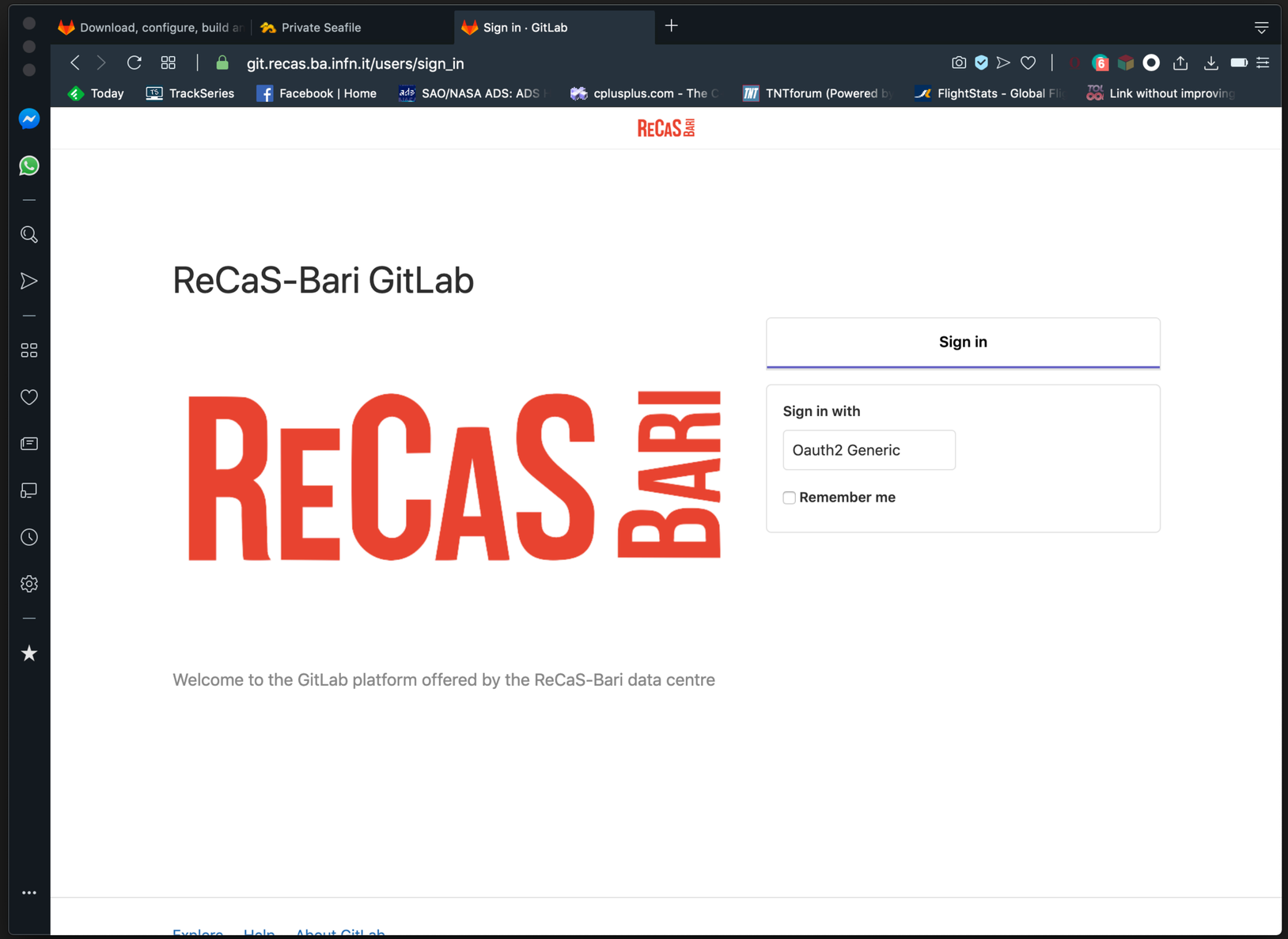
We have configured HerdSoftware to use the GitLab continuous integration facility to compile the code and run the tests automatically on each push. If anything breaks the developer receives an automatically generated email with detailed information on what's gone wrong.

The test jobs also compute the code coverage for the project and give information on which parts of the codebase are not explored by the tests, so that the developers can add the proper tests to the suite and make sure that every line of code is properly tested.



ACCESS TO HERDSOFTWARE

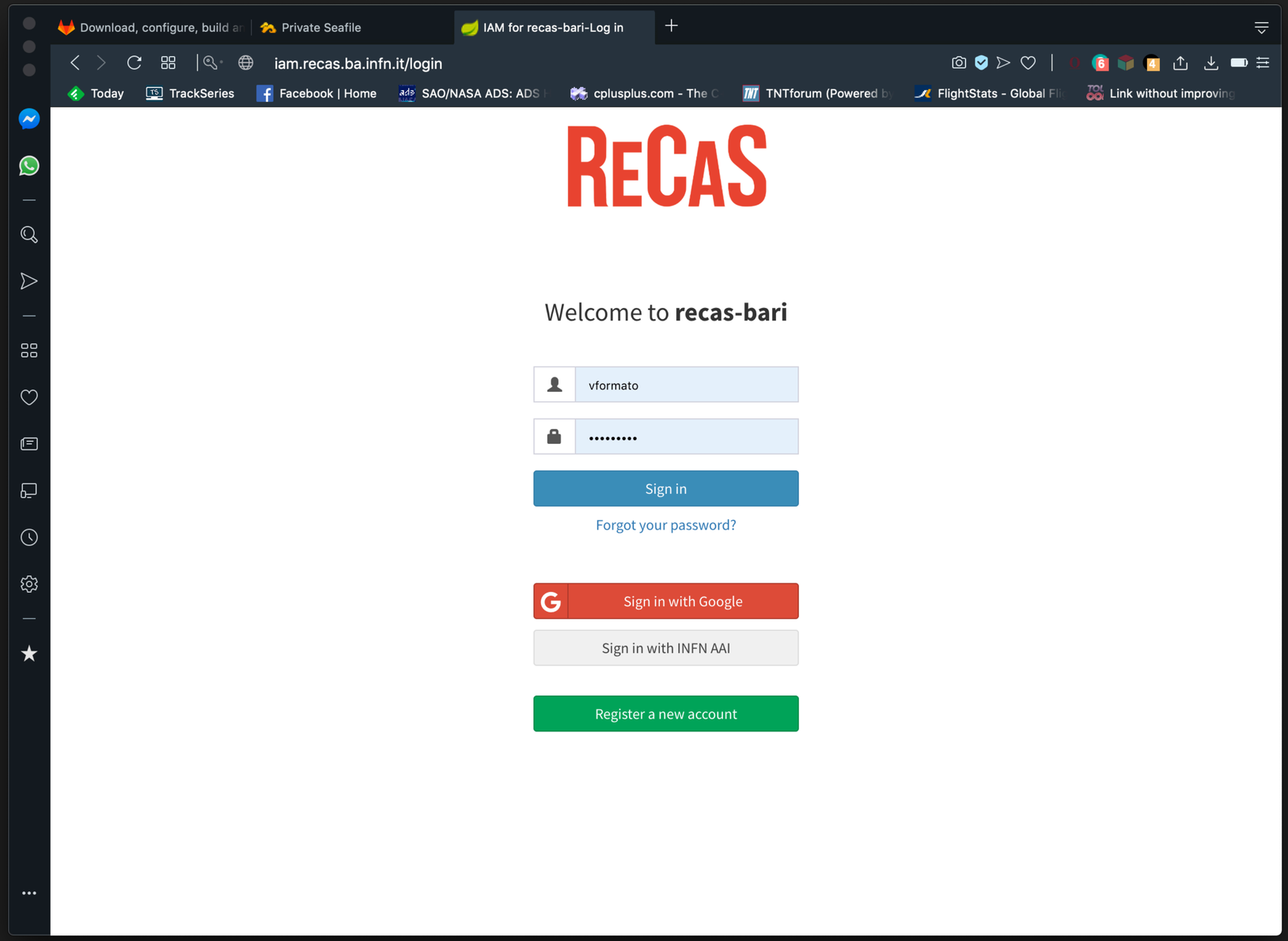
Go to git.recas.ba.infn.it



ACCESS TO HERDSOFTWARE

Go to git.recas.ba.infn.it

You need a INFN or a registered Google account. To register click on the green button.

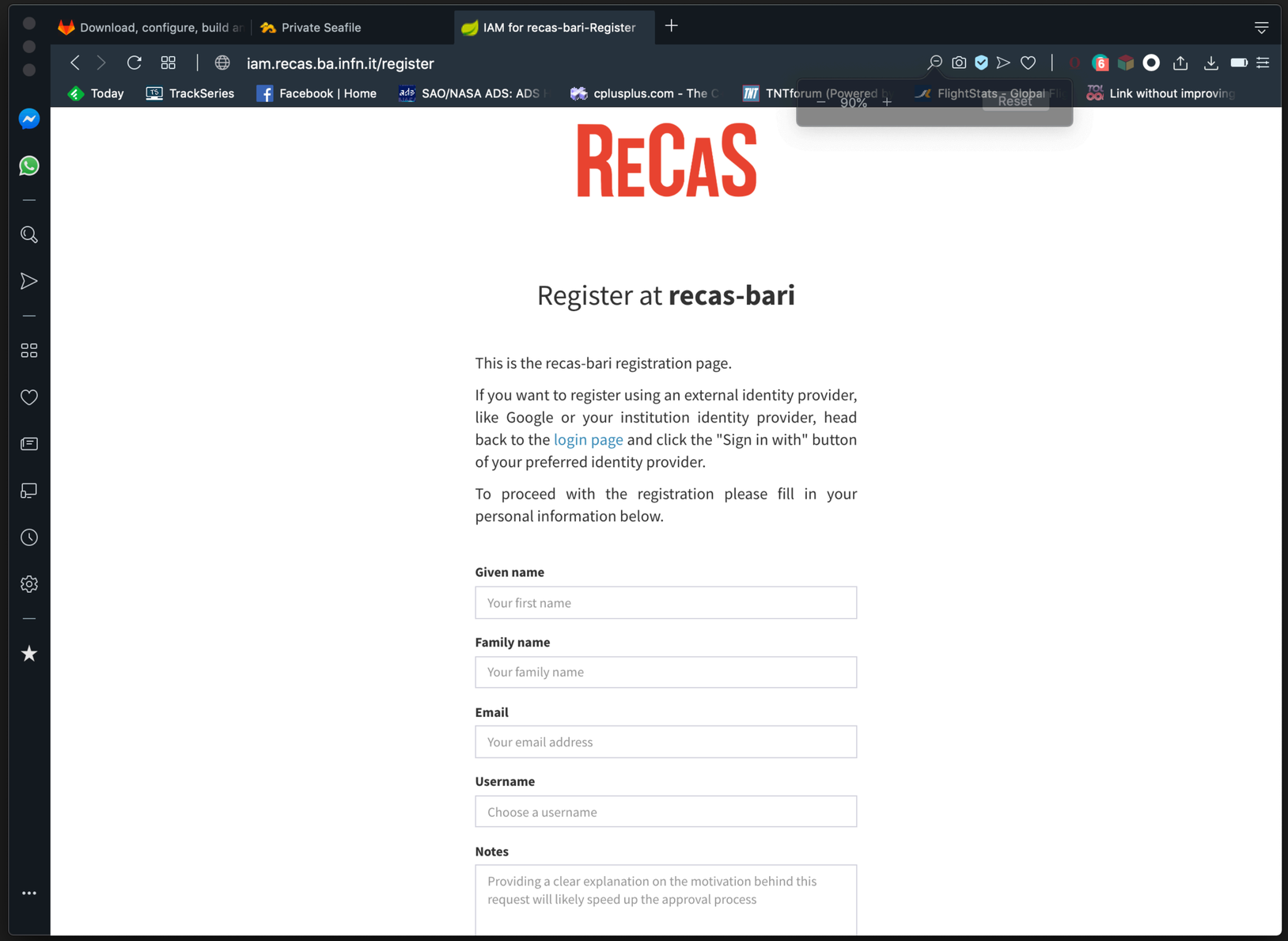


ACCESS TO HERDSOFTWARE

Go to git.recas.ba.infn.it

You need a INFN or a registered Google account. To register click on the green button.

Fill in the details. In the "notes" field type: "Access to the HERD repository (contact person: Fabio Gargano, INFN Bari). Click on "Register".



ACCESS TO HERDSOFTWARE

Go to git.recas.ba.infn.it

You need a INFN or a registered Google account. To register click on the green button.

Fill in the details. In the "notes" field type: "Access to the HERD repository (contact person: Fabio Gargano, INFN Bari). Click on "Register".

After logging, write to one of:

- fabio.gargano@ba.infn.it
- nicola.mori@fi.infn.it
- valerio.formato@roma2.infn.it

to be added to the "herd" group