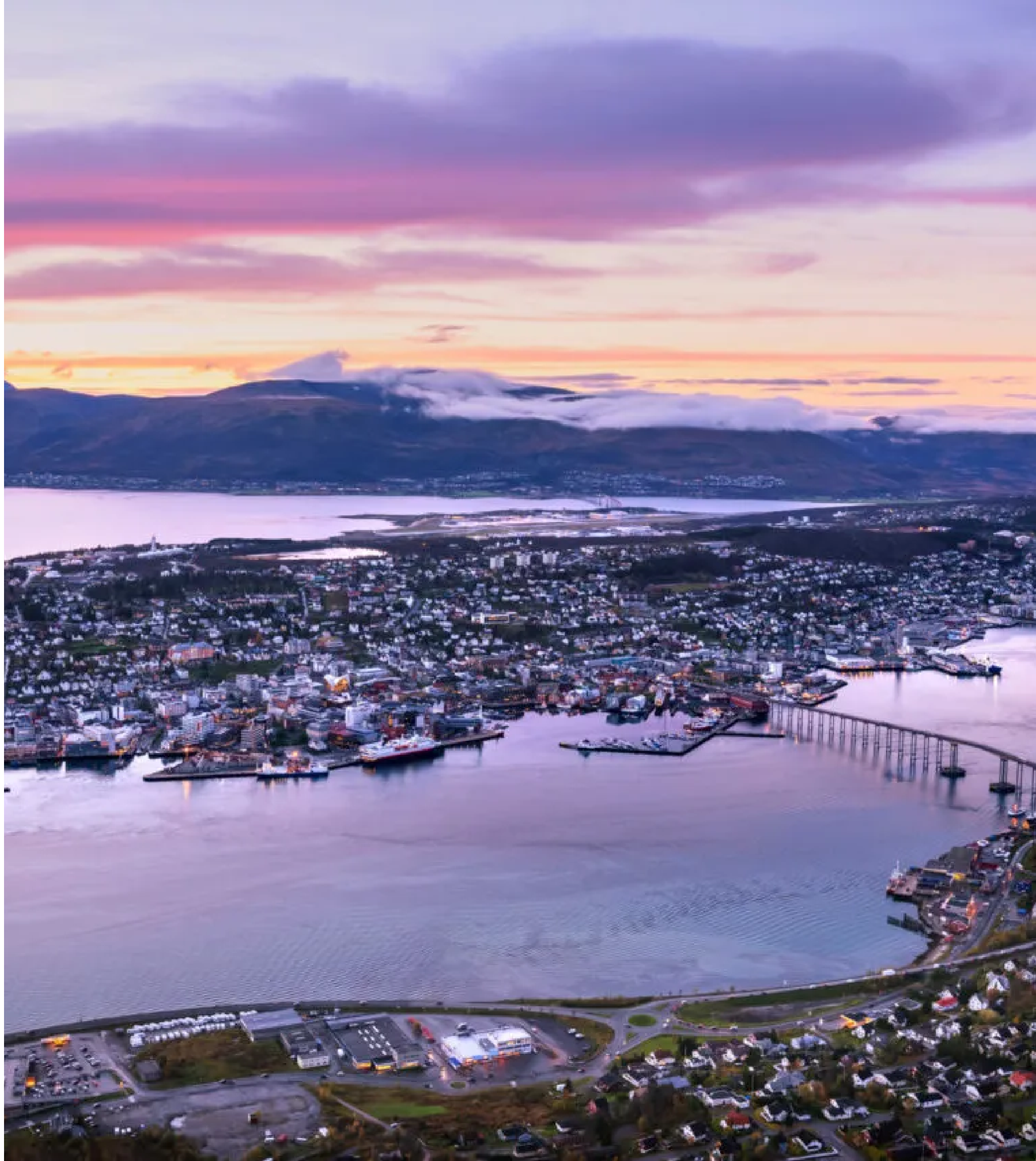
In-house development and collaboration in InPreD-Norge

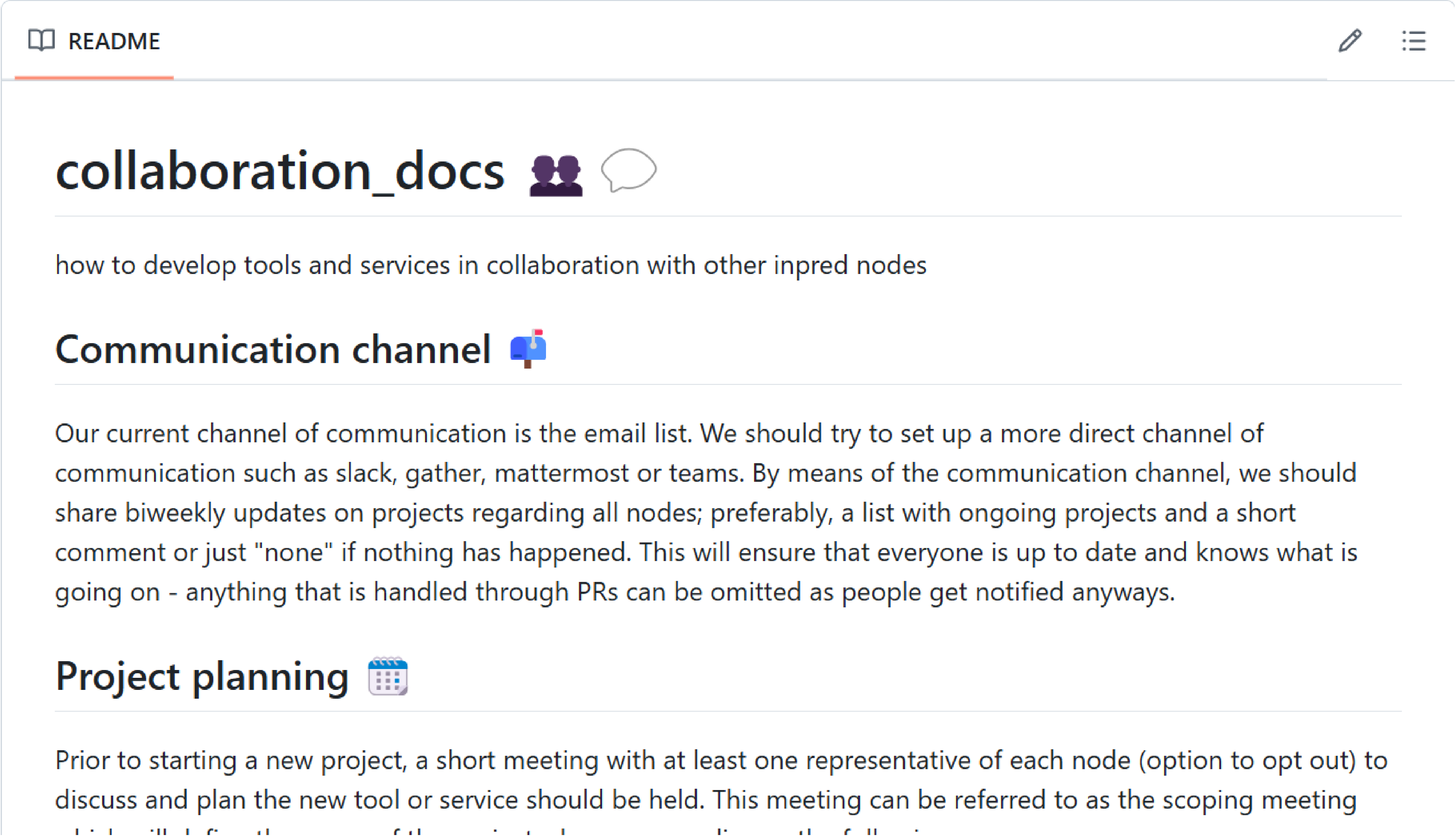
3rd Annual workshop on bioinformatics and variant interpretation in InPreD

https://inpred.github.io/25-06_bioinfo_ws/develop_and_collab



Overview

1. Communication channel
2. Project planning
3. Development
4. Issue and bug handling
5. New features



collaboration_docs 👥 💬

how to develop tools and services in collaboration with other inpred nodes

Communication channel 📫

Our current channel of communication is the email list. We should try to set up a more direct channel of communication such as slack, gather, mattermost or teams. By means of the communication channel, we should share biweekly updates on projects regarding all nodes; preferably, a list with ongoing projects and a short comment or just "none" if nothing has happened. This will ensure that everyone is up to date and knows what is going on - anything that is handled through PRs can be omitted as people get notified anyways.

Project planning 📆

Prior to starting a new project, a short meeting with at least one representative of each node (option to opt out) to discuss and plan the new tool or service should be held. This meeting can be referred to as the scoping meeting

This is a **private repo** at https://github.com/InPreD/collaboration_docs

1. Communication channel

Current situation

- as of today we communicate mainly via email
- some communication via Teams (most of us are "external" lacking some important features features)
- discussions on GitHub via PRs
- we have monthly meetings for updates and discussions

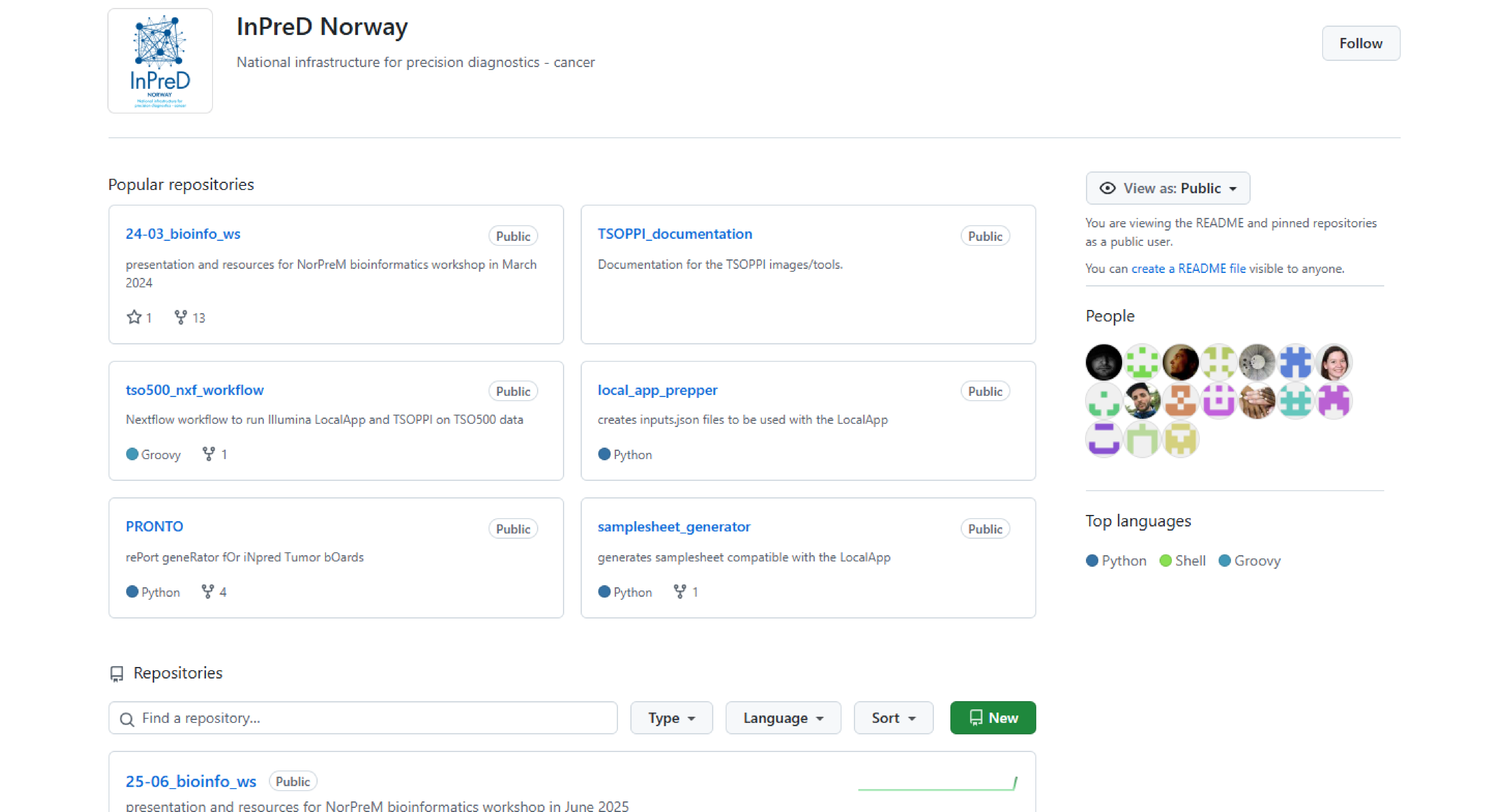
1. Communication channel

Future plans

- requirements for platform/service for communication between nodes:
  - open source
  - easy and safe data sharing between nodes
  - free
  - self-hosted
- include bioinformaticians from clinical genetics departments (some of us are involved in CG already)
- currently, we are testing Zulip and Rocket.Chat
- biweekly updates from all nodes

2. Project Planning

- new projects should be started with a "scoping meeting" were at least one representative of each node
- the following should be discussed and agreed upon:
    - purpose
    - language (default: python)
    - interface (e.g. command line interface, web server)
    - data flow and storage (input and output location, database/filesystem)
    - involved collaborators (which nodes have resources to contribute)
    - deployment options (e.g. baremetal, docker/apptainer)
    - integration with existing projects
    - license (default: GNU AFFERO GENERAL PUBLIC LICENSE - Version 3)
    - intended timeline



InPreD Norway

National infrastructure for precision diagnostics - cancer

Popular repositories

24-03_bioinfo_ws Public

presentation and resources for NorPreM bioinformatics workshop in March 2024

☆ 1 ⑂ 13

TSOPPI_documentation Public

Documentation for the TSOPPI images/tools.

tso500_nxf_workflow Public

Nextflow workflow to run Illumina LocalApp and TSOPPI on TSO500 data

● Groovy ⑂ 1

local_app_prepper Public

creates inputs.json files to be used with the LocalApp

● Python

PRONTO Public

rePort geneRator fOr iNpred Tumor bOards

● Python ⑂ 4

samplesheet_generator Public

generates samplesheet compatible with the LocalApp

● Python ⑂ 1

You are viewing the README and pinned repositories as a public user.

You can create a README file visible to anyone.

People

Top languages

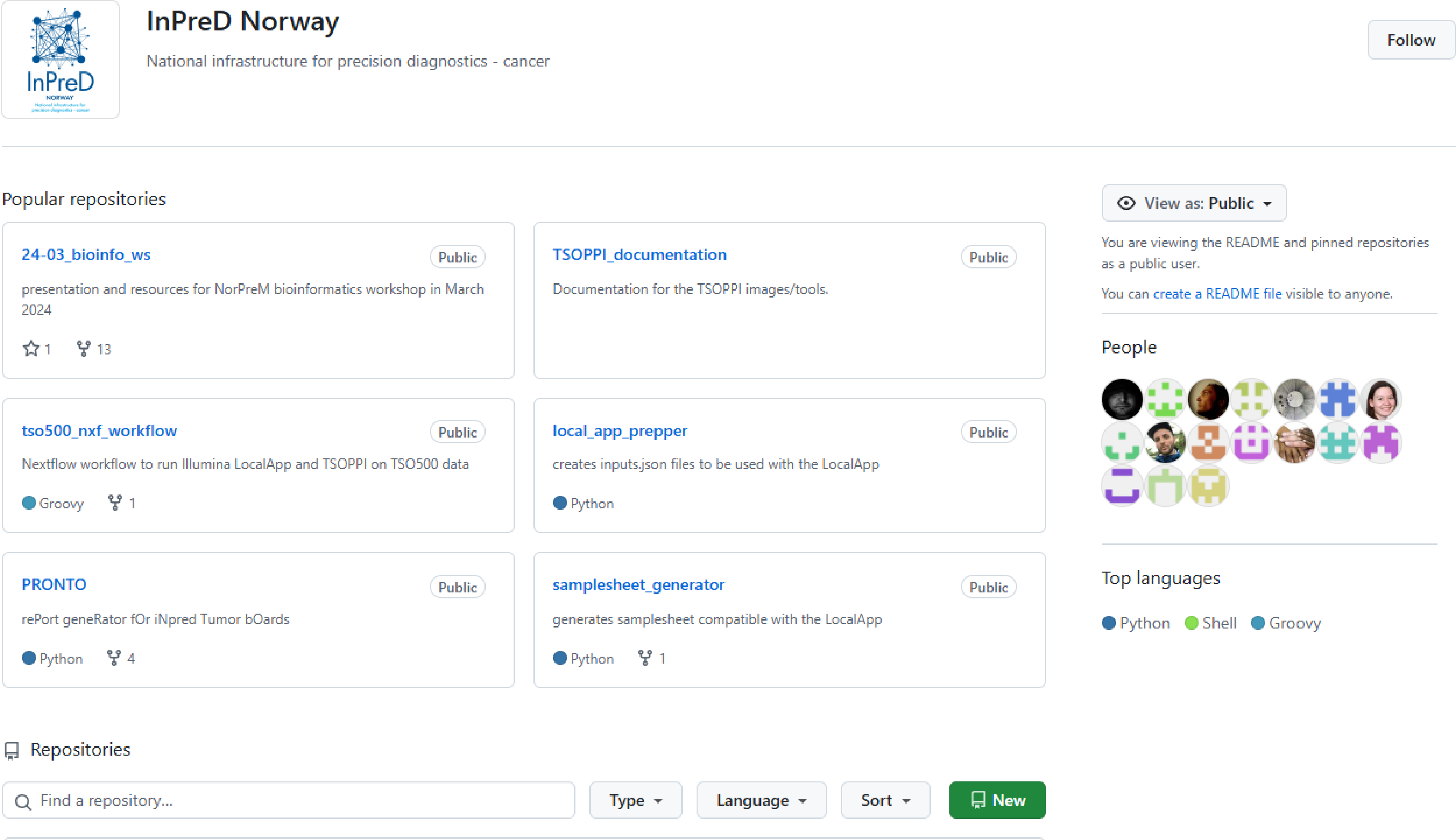
● Python ● Shell ● Groovy

Repositories

25-06_bioinfo_ws Public

presentation and resources for NorPreM bioinformatics workshop in June 2025

- clone it to your local environment and then start developing



InPreD Norway

National infrastructure for precision diagnostics - cancer

Follow

Popular repositories

24-03_bioinfo_ws Public
presentation and resources for NorPreM bioinformatics workshop in March 2024
☆ 1 13

TSOPPI_documentation Public
Documentation for the TSOPPI images/tools.

tso500_nxf_workflow Public
Nextflow workflow to run Illumina LocalApp and TSOPPI on TSO500 data
● Groovy 1

local_app_prepper Public
creates inputs.json files to be used with the LocalApp
● Python

PRONTO Public
rePort geneRator fOr iNpred Tumor bOards
● Python 4

samplesheet_generator Public
generates samplesheet compatible with the LocalApp
● Python 1

View as: Public

You are viewing the README and pinned repositories as a public user.

You can create a README file visible to anyone.

People

Top languages

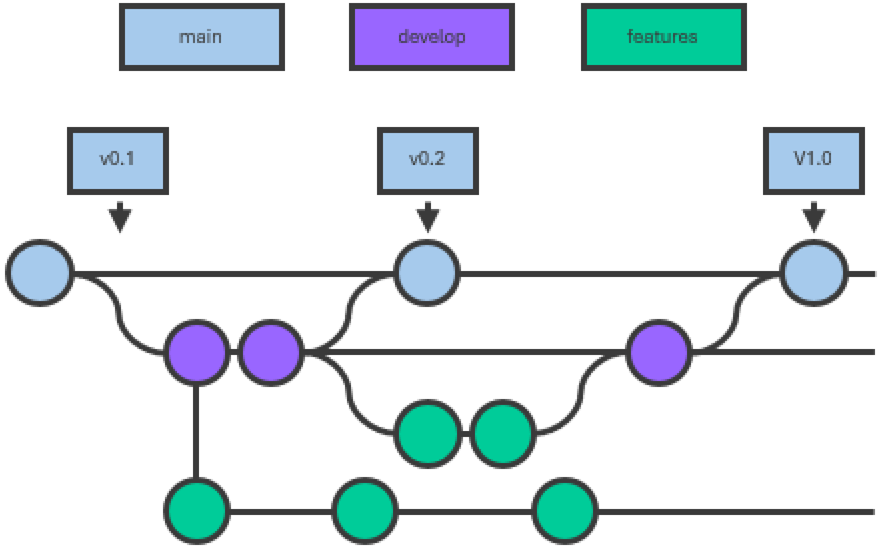
● Python ● Shell ● Groovy

Repositories

Find a repository... Type Language Sort New

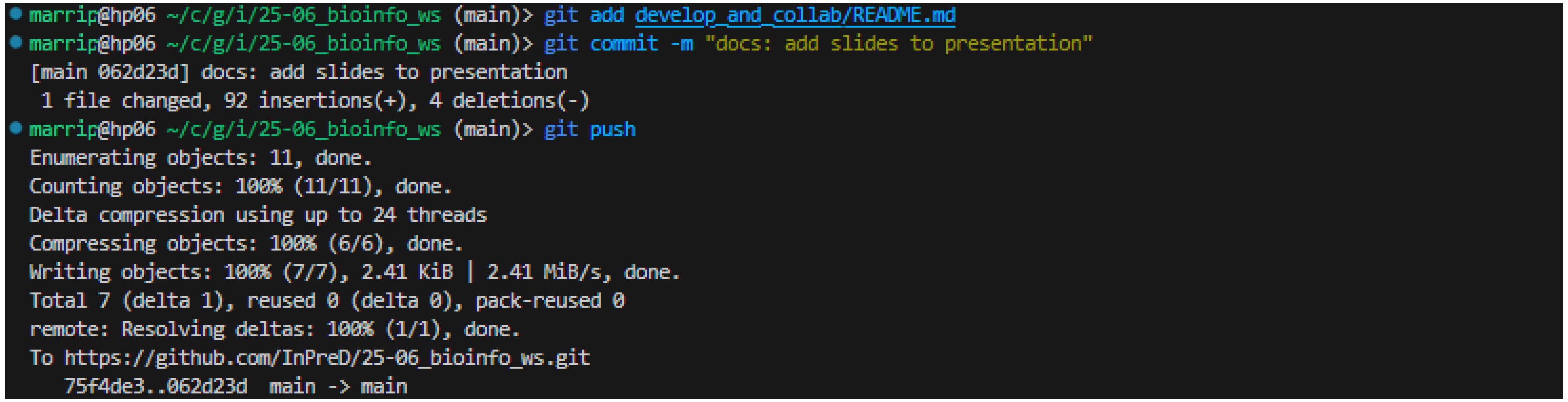
2. Development

3. Use the agreed branching strategy (suggested: simplified Gitflow workflow)



2. Development

4. Commit and push changes early and often to allow others to follow along



```
● marrip@hp06 ~/c/g/i/25-06_bioinfo_ws (main)> git add develop_and_collab/README.md
● marrip@hp06 ~/c/g/i/25-06_bioinfo_ws (main)> git commit -m "docs: add slides to presentation"
  [main 062d23d] docs: add slides to presentation
   1 file changed, 92 insertions(+), 4 deletions(-)
● marrip@hp06 ~/c/g/i/25-06_bioinfo_ws (main)> git push
  Enumerating objects: 11, done.
  Counting objects: 100% (11/11), done.
  Delta compression using up to 24 threads
  Compressing objects: 100% (6/6), done.
  Writing objects: 100% (7/7), 2.41 KiB | 2.41 MiB/s, done.
  Total 7 (delta 1), reused 0 (delta 0), pack-reused 0
  remote: Resolving deltas: 100% (1/1), done.
  To https://github.com/InPreD/25-06_bioinfo_ws.git
     75f4de3..062d23d  main -> main
```

2. Development

5. Follow best practices for the selected programming language

- unit testing (cover test cases from different nodes)
- keeping functions short
- avoid hard-coding
- sensible use of packages and libraries

2. Development

6. Use git commit message conventions

- `feat`, `fix`, `ci`, `test`, `chore`, `docs`, `style`, `perf`, `build`, `refactor`

2. Development

7. Keep the features and PRs small (ideally one PR per feature) to have a tight feedback loop

- focus on one small problem for one feature
- include at least one representative from each node (option to opt out) and set a deadline (e.g. two weeks)

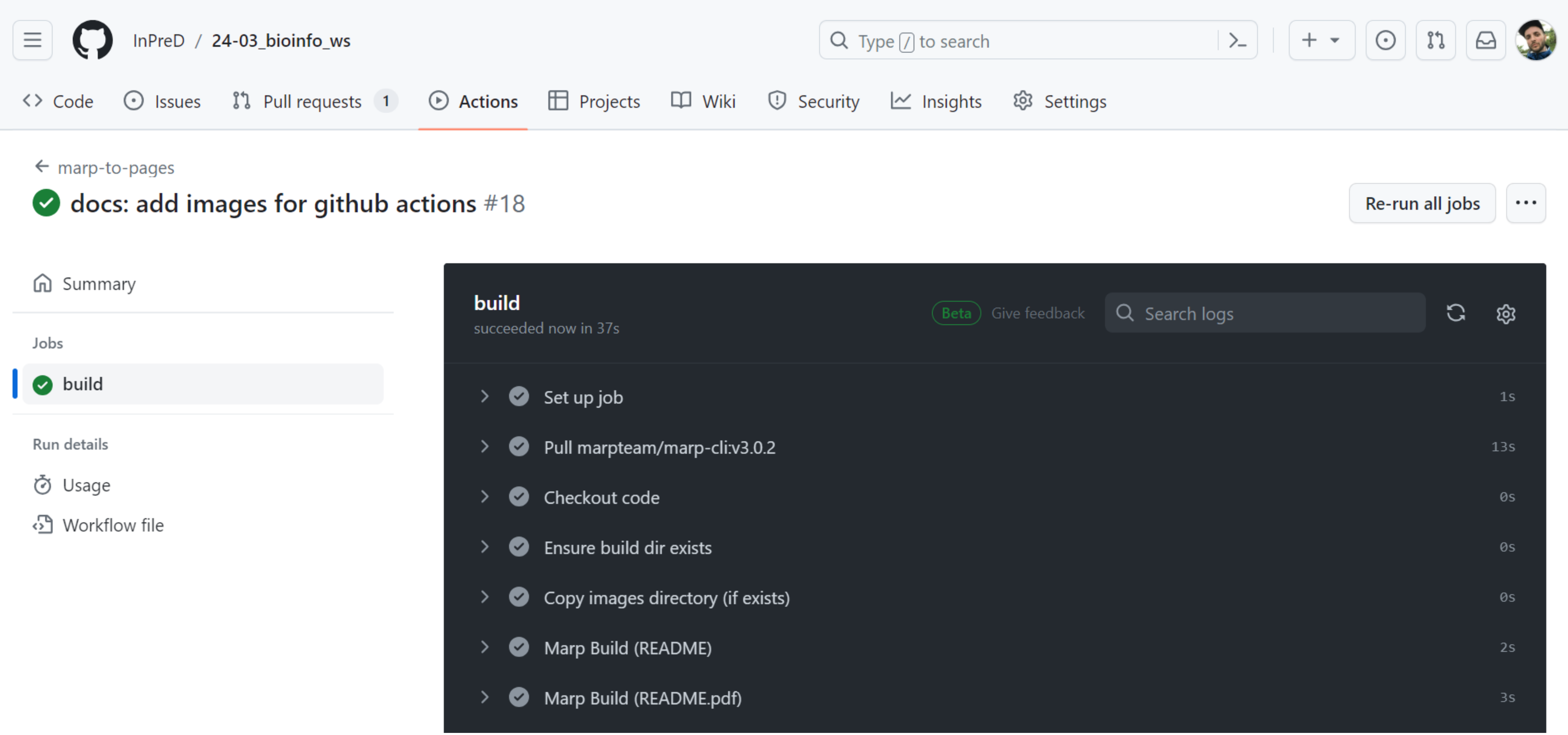
2. Development

8. Pair-programming should be used where it makes sense

- enable knowledge and expertise transfer between the different groups

2. Development

9. Use github actions to test, lint and publish or build your project



docs: add images for github actions #18

build

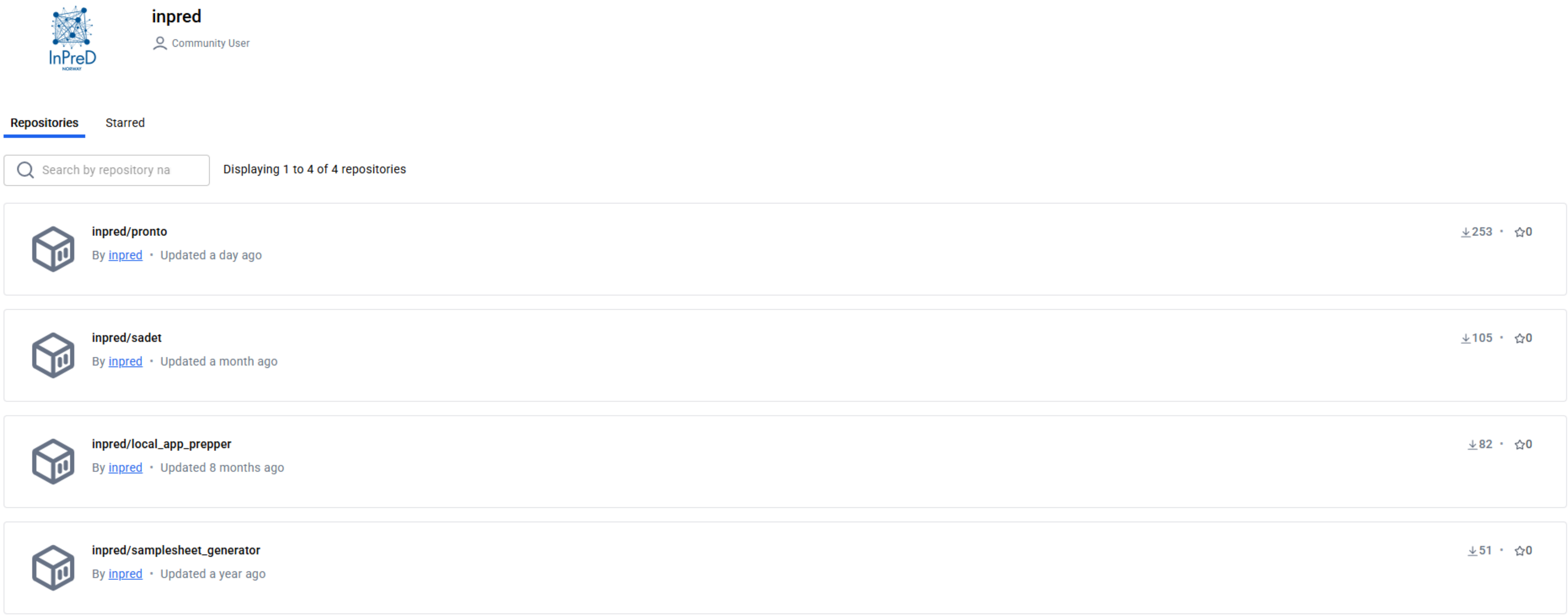
build
succeeded now in 37s

```
Set up job                        1s
Pull marpteam/marp-cli:v3.0.2     13s
Checkout code                     0s
Ensure build dir exists           0s
Copy images directory (if exists) 0s
Marp Build (README)               2s
Marp Build (README.pdf)           3s
```

2. Development

10. Provide at least a docker image (can be converted to apptainer)

- push them to the inpred group at docker hub



inpred

Community User

Displaying 1 to 4 of 4 repositories

inpred/pronto
By inpred · Updated a day ago
↓253 · ☆0

inpred/sadet
By inpred · Updated a month ago
↓105 · ☆0

inpred/local_app_prepper
By inpred · Updated 8 months ago
↓82 · ☆0

inpred/samplesheet_generator
By inpred · Updated a year ago
↓51 · ☆0

2. Development

11. Write documentation and check with others that it is understandable

2. Development

12. Tag and release code that is ready for production using semantic versioning

- `MAJOR`.`MINOR`.`PATCH`