

# Bioinformatics session

A two-day workshop for  
bioinformaticians and molecular  
biologists with focus on the TSO500  
pipeline in InPreD



# Overview

1. Setup
2. Development & Collaboration
3. Nextflow
4. tso500\_nxf\_workflow
5. Python

# 1. Setup

# Create a GitHub account

- go to <https://github.com/> and click on `Sign up`



# Create a GitHub account

- enter your email

Welcome to GitHub!

Let's begin the adventure

**Enter your email\***

→ coder@inpred.no

Continue

# Create a GitHub account

- set a password

Welcome to GitHub!

Let's begin the adventure

Enter your email\*

✓ coder@inpred.no

Create a password\*

→ ●●●●●●●●



Continue

# Create a GitHub account

- choose a username

Welcome to GitHub!

Let's begin the adventure

Enter your email\*

✓ coder@inpred.no

Create a password\*

✓ ••••••••••

Enter a username\*

→ inpredder

Continue

# Create a GitHub account

- choose email preferences

## Email preferences

☐ Receive occasional product updates and announcements.

Continue



# Create a GitHub account

- solve the puzzle



# Create a GitHub account

- create your account



# Create a GitHub account

- find the activation code in the email you received



Here's your GitHub launch code, @inpredder!



Continue signing up for GitHub by entering the code below:

40619601

Open GitHub

# Create a GitHub account

- select the desired options

This will help us guide you to the tools that are best suited for your projects.

How many team members will be working with you?

☒ Just me

☐ 2-5

☐ 5-10

☐ 10-20

☐ 20-50

☐ 50+

Are you a student or teacher?

☒ N/A

☐ Student

☐ Teacher

Continue

## What specific features are you interested in using?

Select all that apply so we can point you to the right GitHub plan.



Collaborative coding

Codespaces, Pull requests, Notifications, Code review, Code review assignments, Code owners, Draft pull requests, Protected branches, and more.



Automation and CI/CD

Actions, Packages, APIs, GitHub Pages, GitHub Marketplace, Webhooks, Hosted runners, Self-hosted runners, Secrets management, and more.

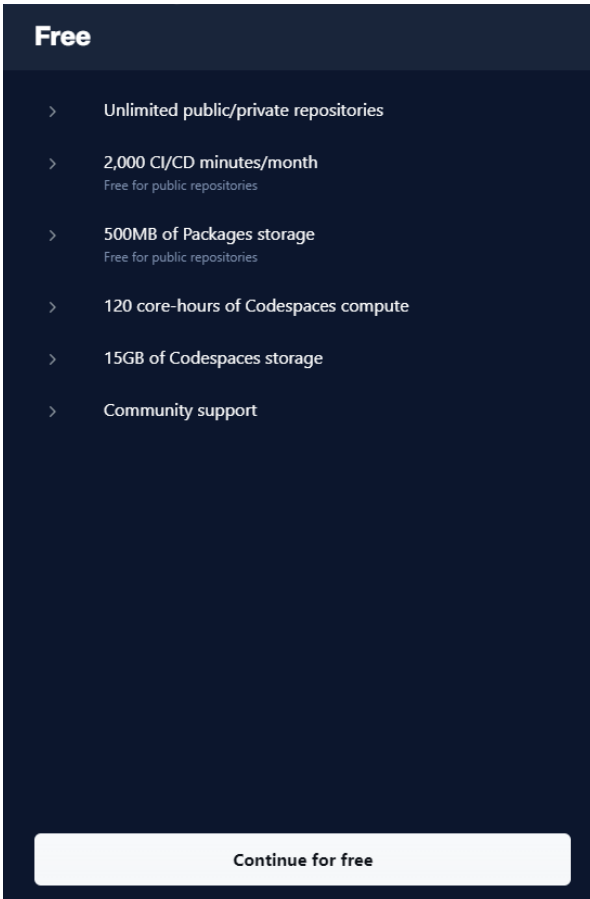


Security

Private repos, 2FA, Required reviews, Required status checks, Code scanning, Secret scanning, Dependency graph, Dependabot alerts, and more.

# Create a GitHub account

- choose the free plan



The screenshot shows the GitHub Free plan selection screen. It features a dark blue background with a list of features for the Free plan. At the bottom, there is a white button labeled "Continue for free".

**Free**

- > Unlimited public/private repositories
- > 2,000 CI/CD minutes/month  
Free for public repositories
- > 500MB of Packages storage  
Free for public repositories
- > 120 core-hours of Codespaces compute
- > 15GB of Codespaces storage
- > Community support

[Continue for free](#)

# **1. Setup**

**Be added to InPreD organisation at GitHub**

# 1. Resources

- [Getting started with your GitHub account](#)

## 2. Development & Collaboration

### Short `git` introduction

- distributed version control system
- tracks history of changes committed by different contributors
- every developer has full copy of project and its history



## Short `git` introduction

### Basic `git` commands

`git init` : initialises new git repository

`git clone <repository url>` : creates local copy of remote repository

`git add <file/s>` : stage new or changed files (anything that should be committed to the repository)

`git commit -m "feat: my new feature"` : commit changes to the repository

## Basic `git` commands

### commit message conventions

`<type>[optional scope]: <description>`

- `feat` : new feature
- `fix` : patching bug
- `refactor` : code change that neither is neither feat nor fix
- `build` : build system related changes
- `perf` : improving performance

## commit message conventions

`<type>[optional scope]: <description>`

- `chore` : code unrelated changes, e.g. dependencies
- `style` : code change that does not change meaning
- `test` : changes to tests
- `docs` : adding/updating documentation
- `ci` : continuous integration, e.g. github actions

## Basic `git` commands

`git status` : overview over untracked, modified and staged changes

`git branch` : show local branches

`git merge` : merge branches

`git pull` : load changes from remote counterpart

`git push` : upload changes to remote counterpart

## **Branching strategy - Gitflow**

- github actions (linting, testing, building)
- pull requests (best practice)
- release and semantic versioning
- licensing

## 2. Resources

- [About Git](#)
- [Gitflow workflow](#)

### 3. Nextflow

- general (install, best practice)
- nf-core template
- stubbing



## 4. `tso500_nxf_workflow`

- status
- demonstration

## 5. Python

- general (best practice, cli)
- unit testing (pytest)

## Resources

- [github actions](#)
- [nf-core](#)
- [pytest unittesting](#)
- [semantic versioning](#)