Software Lifecycle Management

Final Project

Java Game TicTacToe

by Eva Lehner, Florian Scholz, Sophie Schönwälder-Rieder
February, 2026

Table of Contents

- 000 - Welcome
- 001 - Create a GitHub repository (public)
- 002 - Create a GitHub project and link it to the created repository
- 003 - Transfer the user stories to a new Kanban board in the project
- 004 - Add additional issues to the kanban board
- 005 - Create a new Java Maven project
- 006 - Set the GitHub repository as a remote repository
- 007 - Commit and push your blank Java project to the main branch
- 008 - Protect the main branch
- 009 - CI_CD Scripts and workflows
- 010 - Working on User Stories
- 011 - A final overview

000 - Welcome

This document serves as documentation for the SLM-project 'Java Game TicTacToe' and provides a walkthrough illustrating the most important steps of our development process with short textual description as well as screenshots - focussing on our usage of GitHub and application of concepts studied in the seminar rather than on the game itself.

The structure of the documentation follows the chronology of our workflow - which in turn closely followed the list of tasks provided in the task description.

Regarding our collaborative process: Especially in the earlier stages of the project, the majority of tasks were done together, triple programming style. The intent was to maximise knowledge gain for all team members.

This is mentioned here to prevent faulty impressions that might be caused by the statistics showing that one member of the team authored significantly more commits than the other two members in the beginning. This is due to the fact that their account was typically used for the tasks we tackled together, mainly to simplify the workflow and keep the screenshots similar.

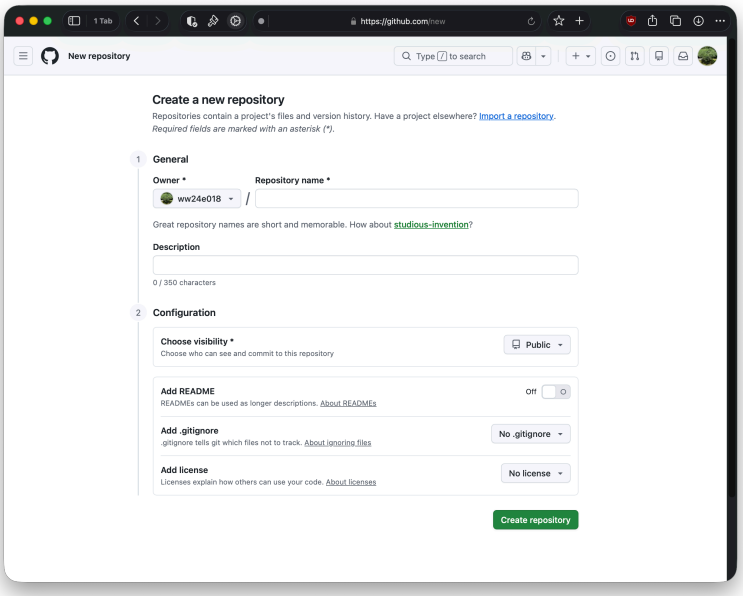
The GitHub repository can be accessed at https://github.com/ww24e018/slm-tictactoe
The project (Kanban Board) can be accessed at
https://github.com/users/ww24e018/projects/1
The .jar of our latest release of the game can be downloaded at
https://github.com/ww24e018/slm-tictactoe/releases

In the process of exploring workflows and ci/cd, a test repository was created to provide a playground for quick trials that require changes to be present on the main branch. This repo can be accessed here, should it be of any interest: https://github.com/ww24e018/slm-tictactoe-test

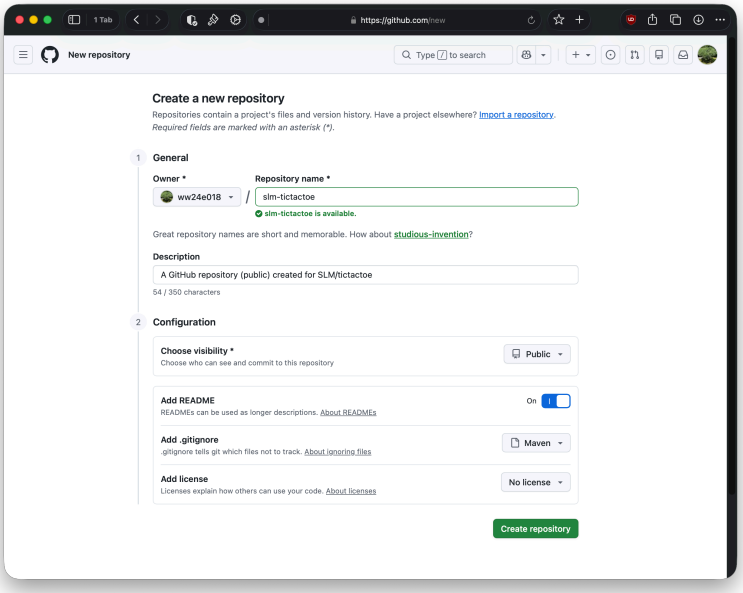
Contributors: Eva Lehner, Florian Scholz, Sophie Schönwälder-Rieder

001 - Create a GitHub repository (public)

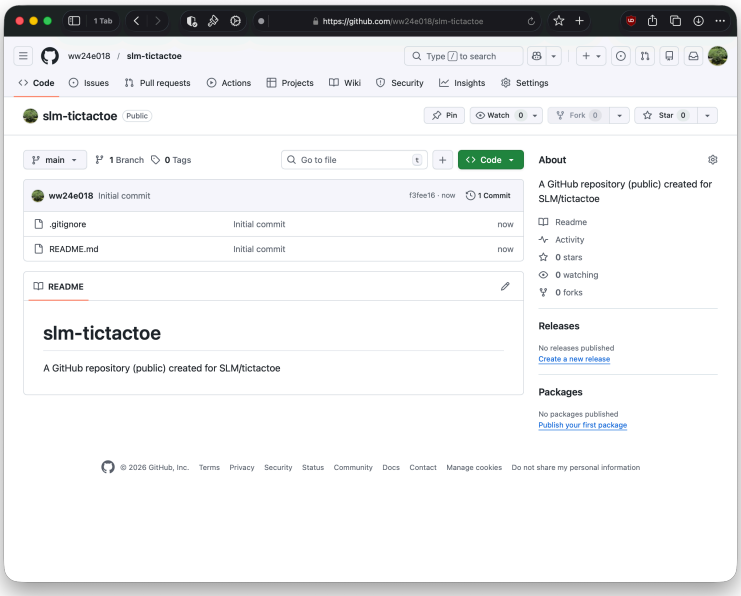
At https://github.com/repos a new repository was created by clicking "new repository":



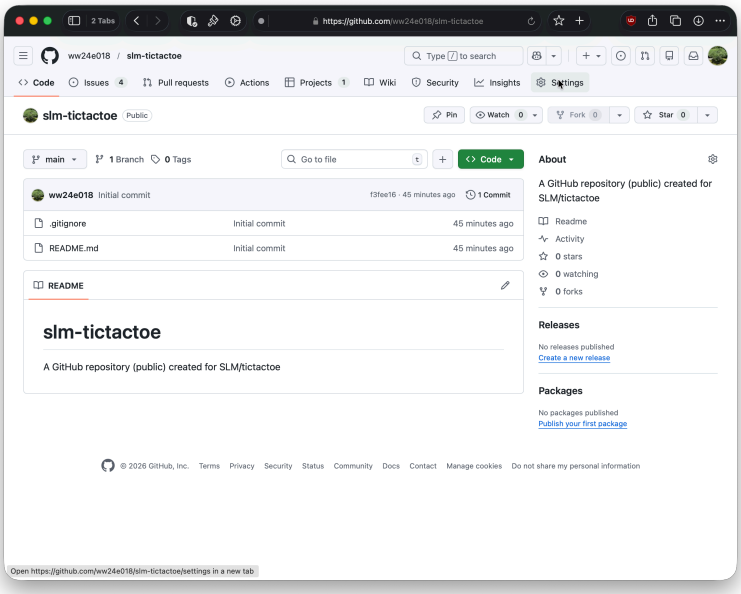
The name "slm-tictactoe" was chosen.



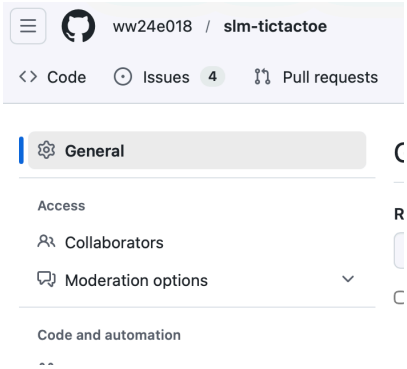
The repository was successfully created.

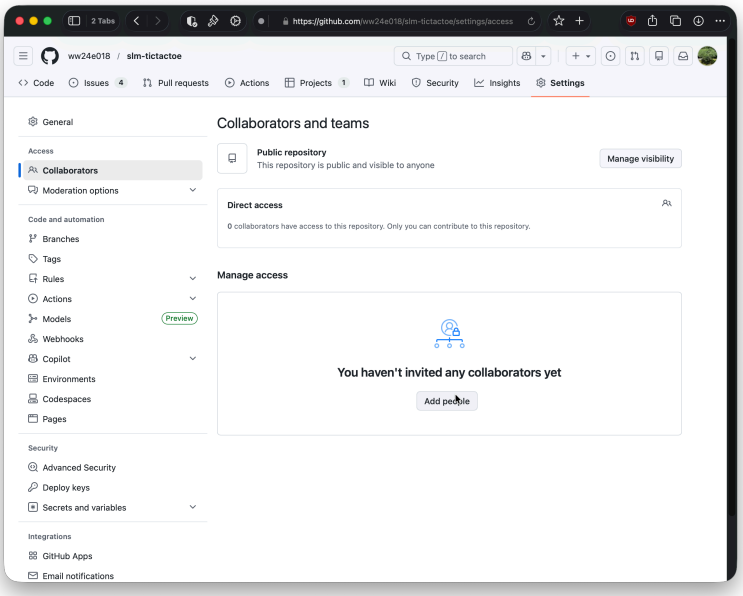


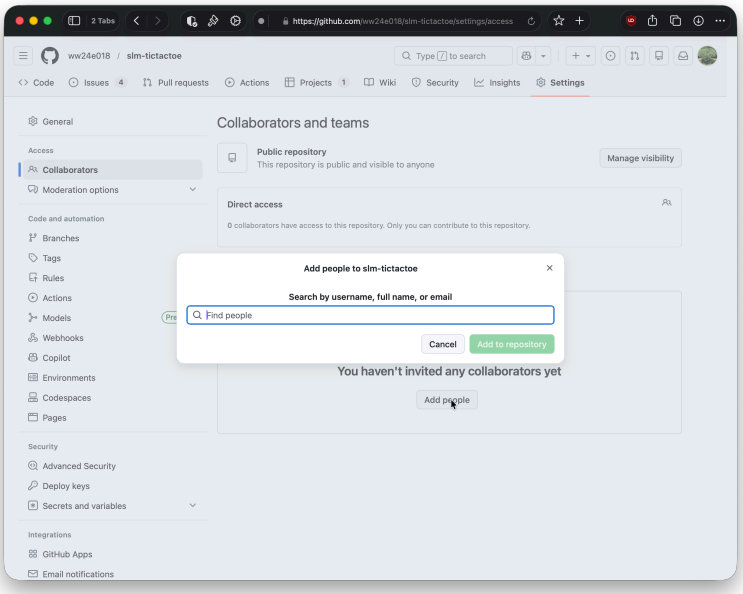
In order to successfully collaborate in this new repository, the other team members were invited as collaborators:
Go to "Settings":

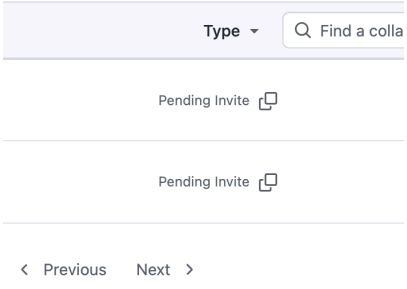


Go to "Collaborators":



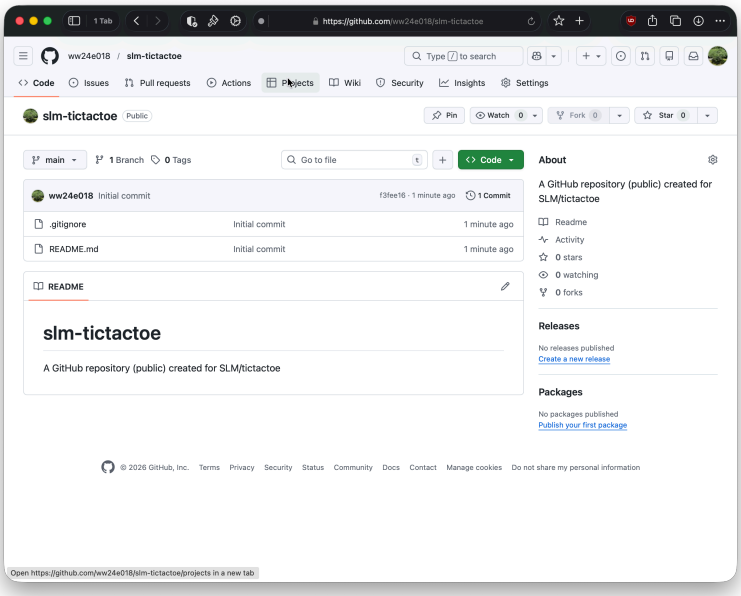

https://github.com/ww24e018/slm-tictactoe/settings/access
ww24e018 / slm-tictactoe
Type / to search
Code
Issues 4
Pull requests
Actions
Projects 1
Wiki
Security
Insights
Settings
General
Access
Collaborators
Moderation options
Code and automation
Branches
Tags
Rules
Actions
Models
Preview
Webhooks
Copilot
Environments
Codespaces
Pages
Security
Advanced Security
Deploy keys
Secrets and variables
Integrations
GitHub Apps
Email notifications
Collaborators and teams
Public repository
This repository is public and visible to anyone
Manage visibility
Direct access
0 collaborators have access to this repository. Only you can contribute to this repository.
Manage access
You haven't invited any collaborators yet
Add people

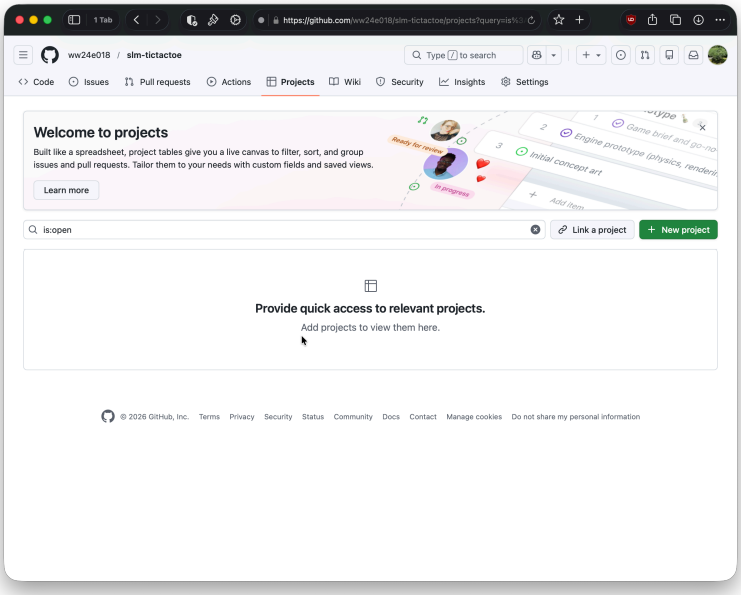

https://github.com/ww24e018/slm-tictactoe/settings/access
ww24e018 / slm-tictactoe
Type / to search
Code
Issues 4
Pull requests
Actions
Projects 1
Wiki
Security
Insights
Settings
General
Access
Collaborators
Moderation options
Code and automation
Branches
Tags
Rules
Actions
Models
Webhooks
Copilot
Environments
Codespaces
Pages
Security
Advanced Security
Deploy keys
Secrets and variables
Integrations
GitHub Apps
Email notifications
Collaborators and teams
Public repository
This repository is public and visible to anyone
Manage visibility
Direct access
0 collaborators have access to this repository. Only you can contribute to this repository.
Add people to slm-tictactoe
Search by username, full name, or email
Find people
Cancel
Add to repository
You haven't invited any collaborators yet
Add people


Type
Find a colla
Pending Invite
Pending Invite
Previous
Next

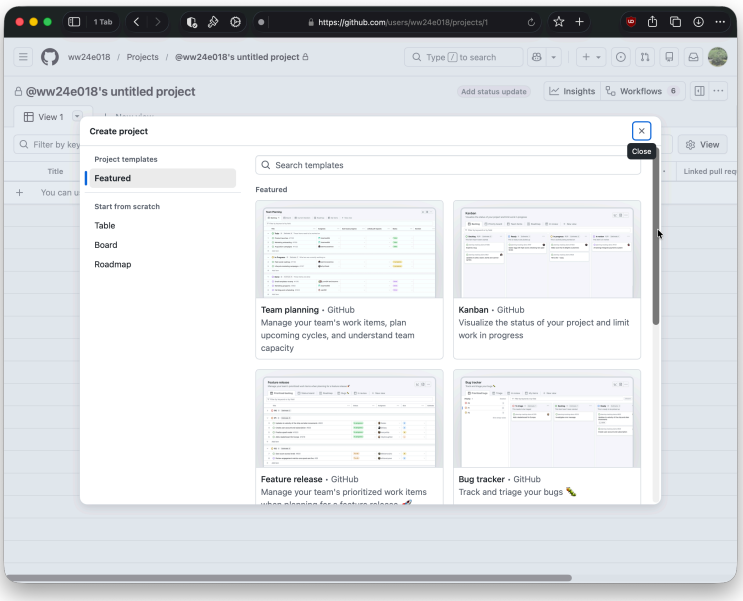
002 - Create a GitHub project and link it to the created repository

A new project was created by choosing the tab "Projects":

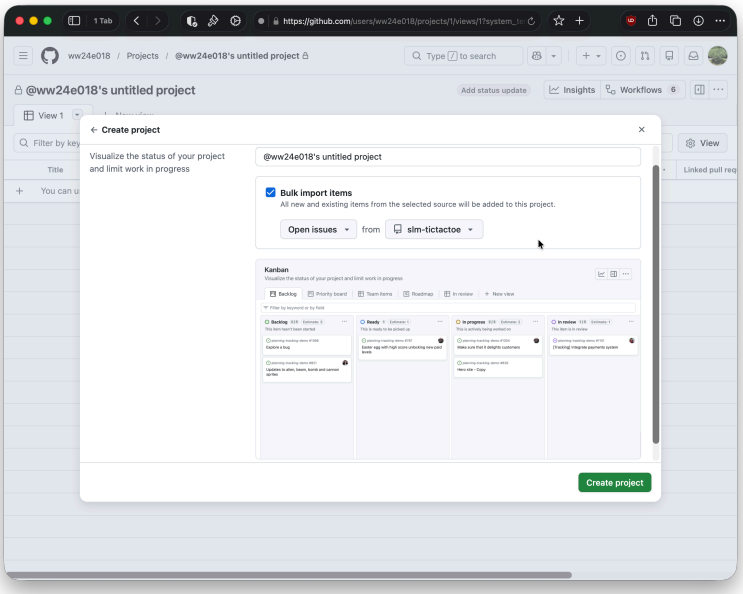




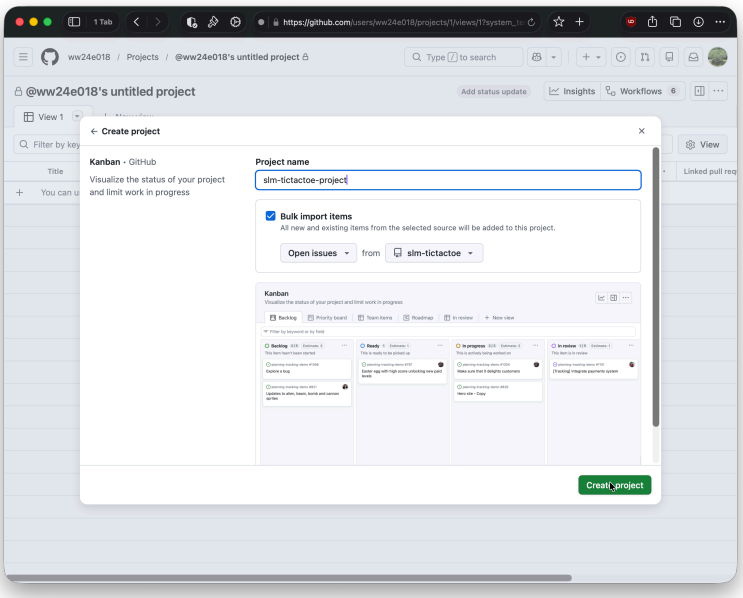
After clicking "+ New project":



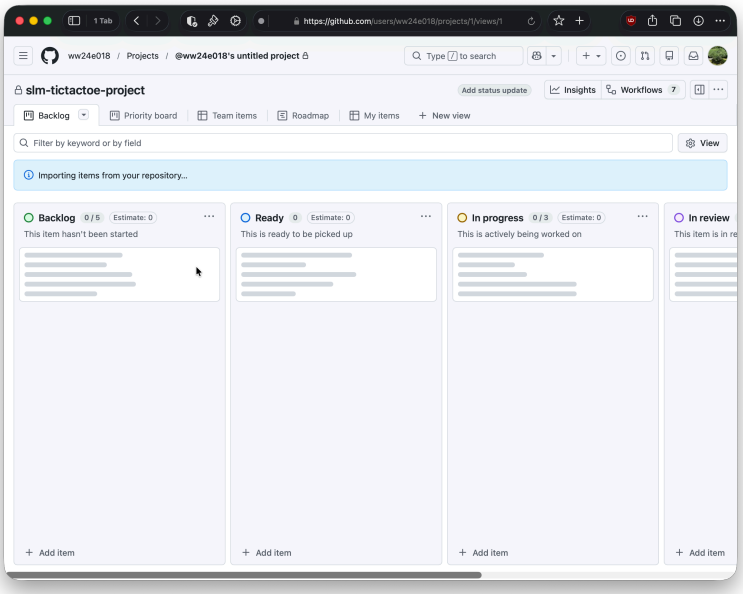
Choosing "Kanban":



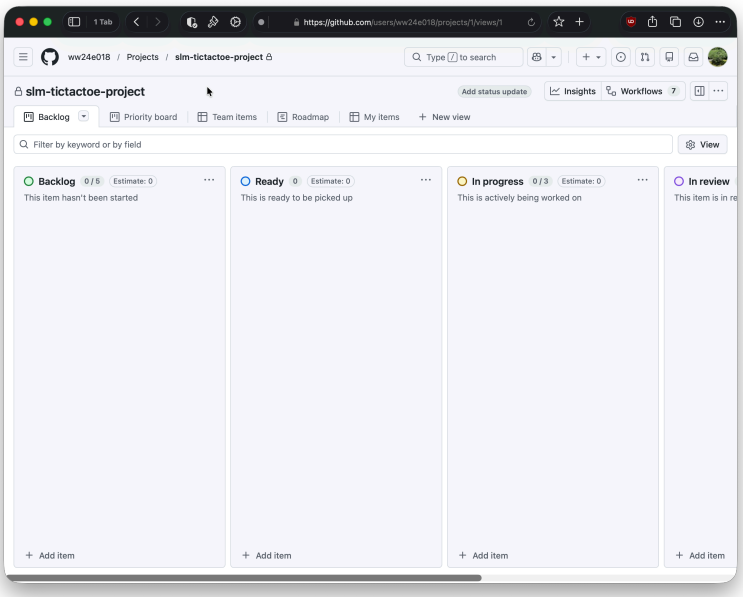
Naming new project and linking it to our existing repository:



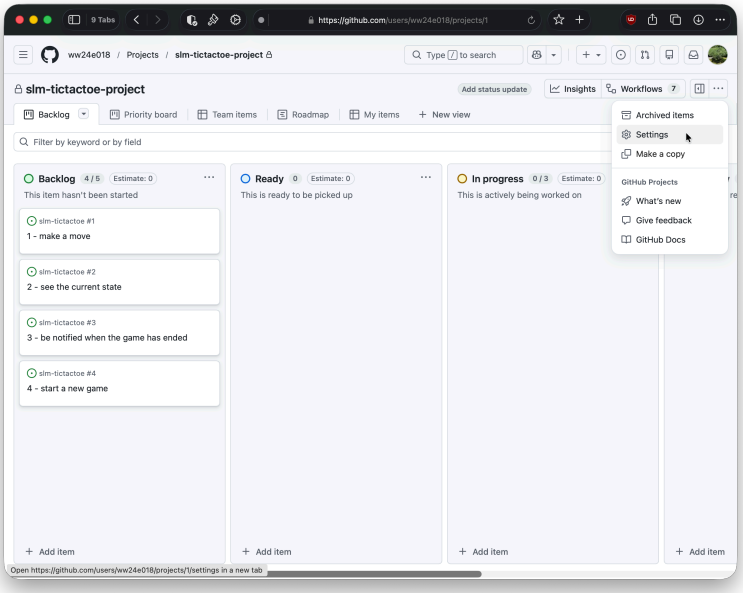
Hooray! We have a blank kanban board!



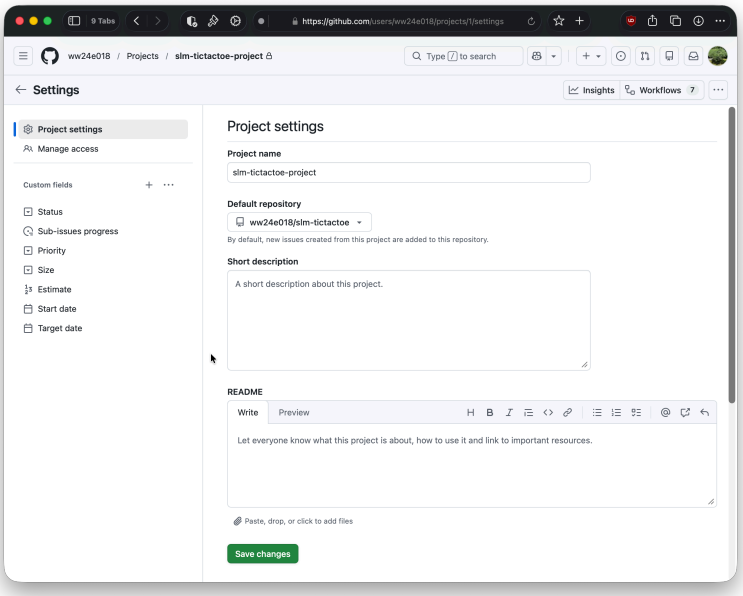
after reload:



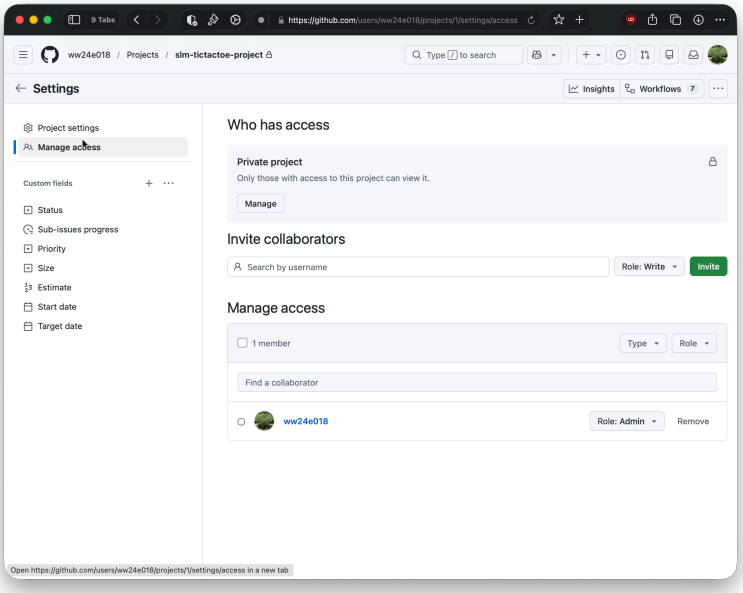
At first we assumed that rights to the repo automatically give rights to the project linked to it. After realising that this was not the case, all team members were invited to the project as well:



clicking Settings:



Clicking "Manage access":



Inviting via "Invite" via the process followed.

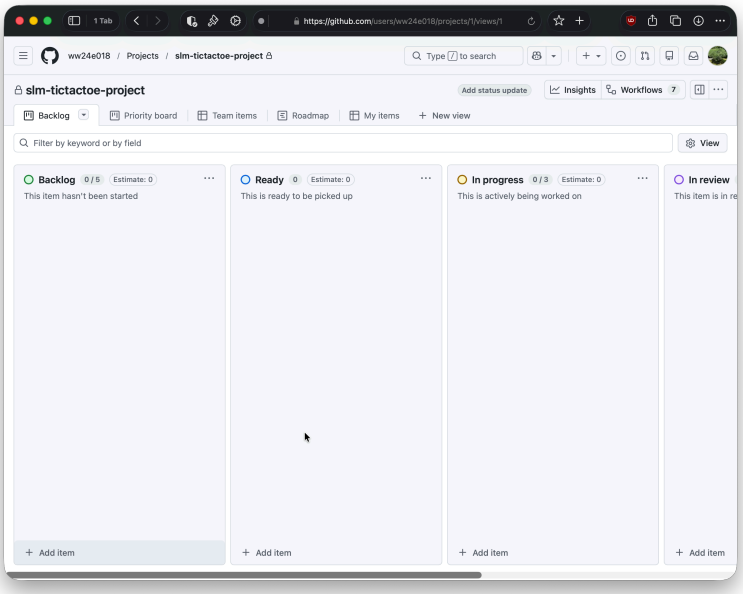
003 - Transfer the user stories to a new Kanban board in the project

The user stories in the project description are:

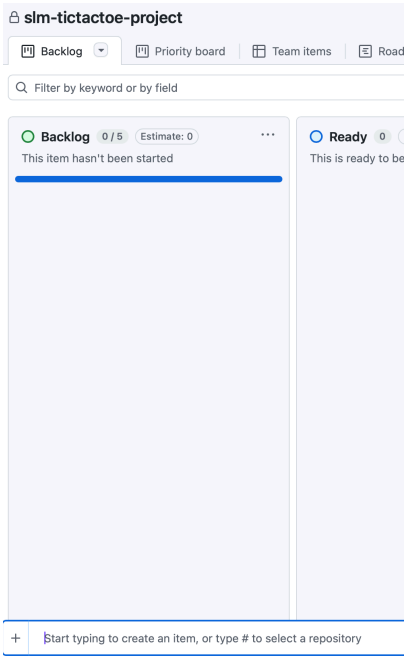
- As a player, I want to be able to **make a move** by choosing an empty square, so that I can place my symbol on the board. (1)
- As a player, I want to be able to **see the current state** of the game, so that I can keep track of the moves made by both myself and my opponent. (2)
- As a player, I want to **be notified when the game has ended** in a win, loss or draw, so that I can see the result of the game. (3)
- As a player, I want to be able to **start a new game** after the current game has ended, so that I can play again. (4)

The bold parts will be chosen as titles. The parts in () are the Ids given by us.

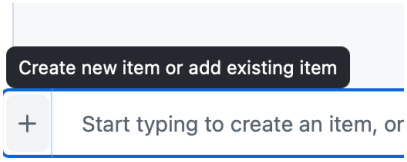
The Board:

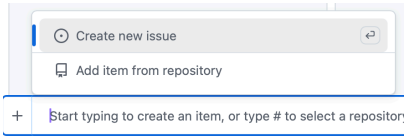


Clicking "add item" in the backlog-column:

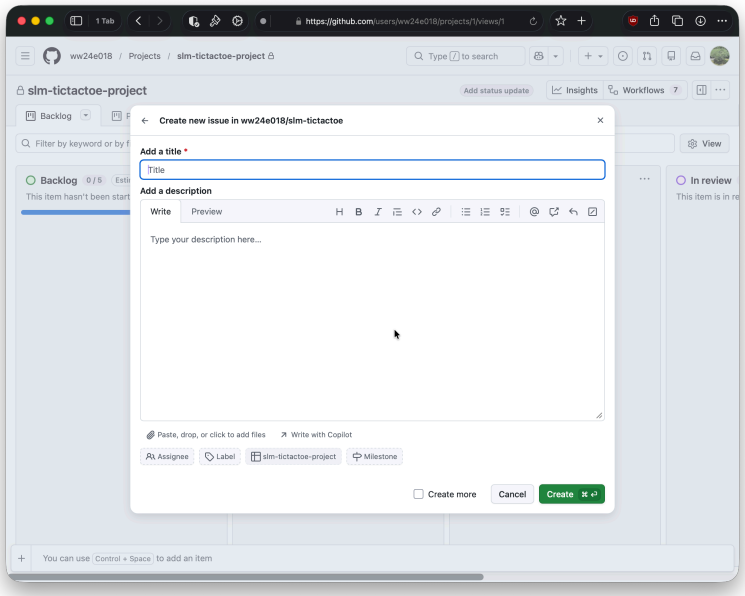


Clicking "+" is possible:

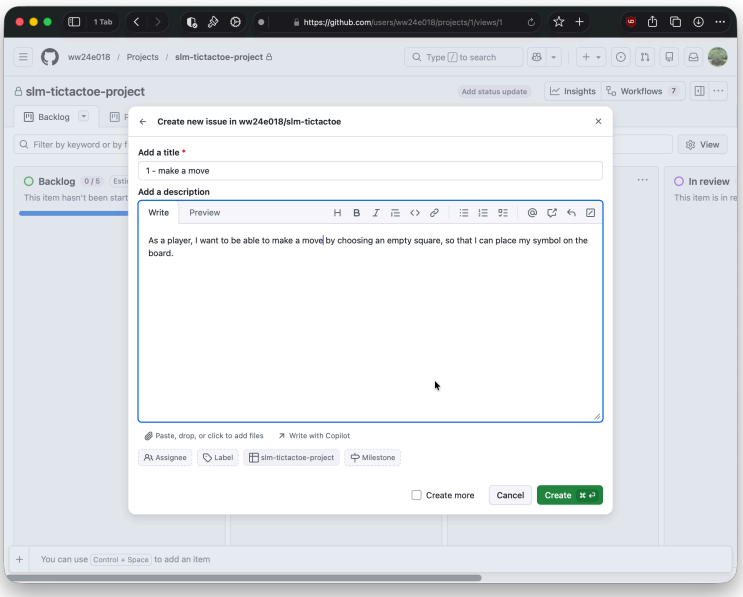




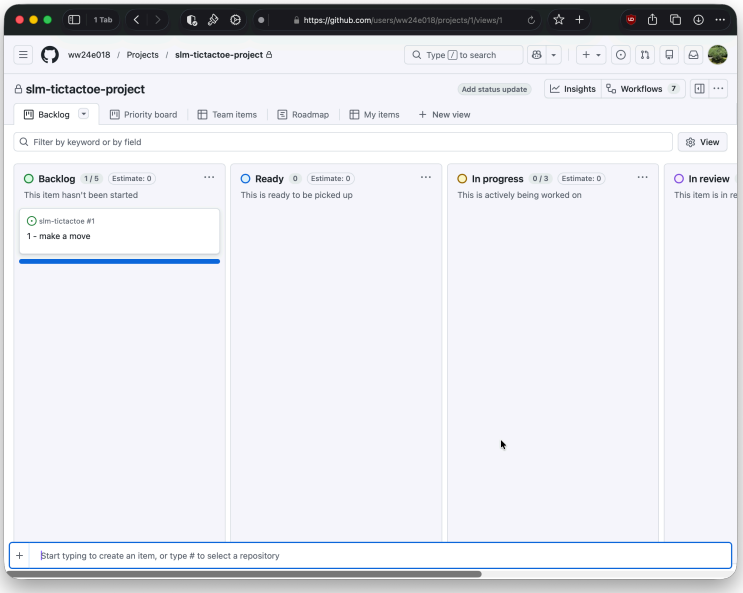
`create new issue`



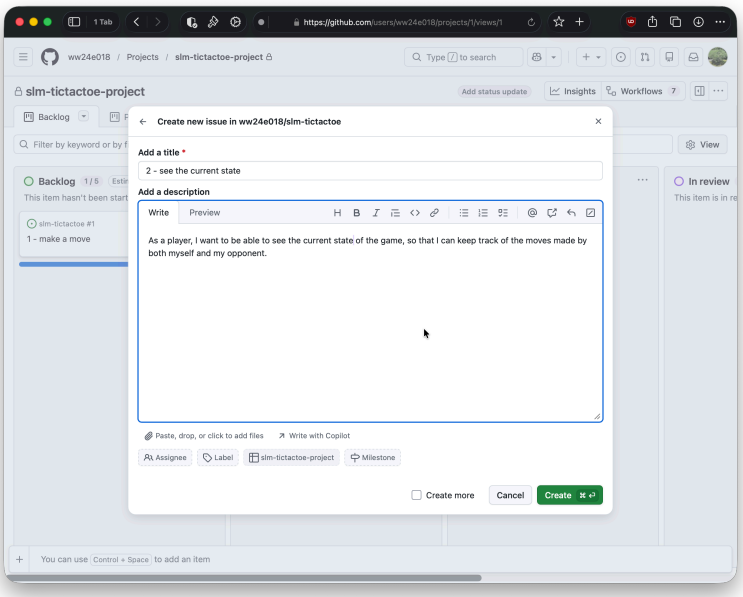
Filling in title and description:

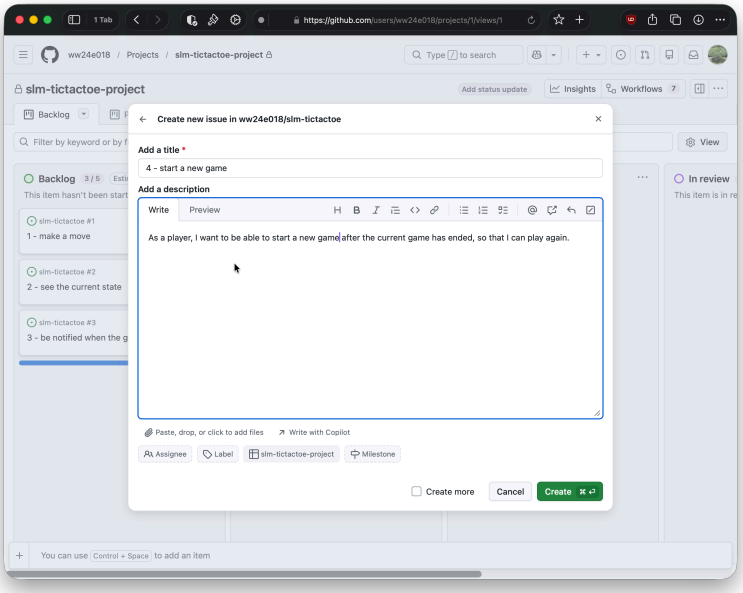


The first issue has been created:

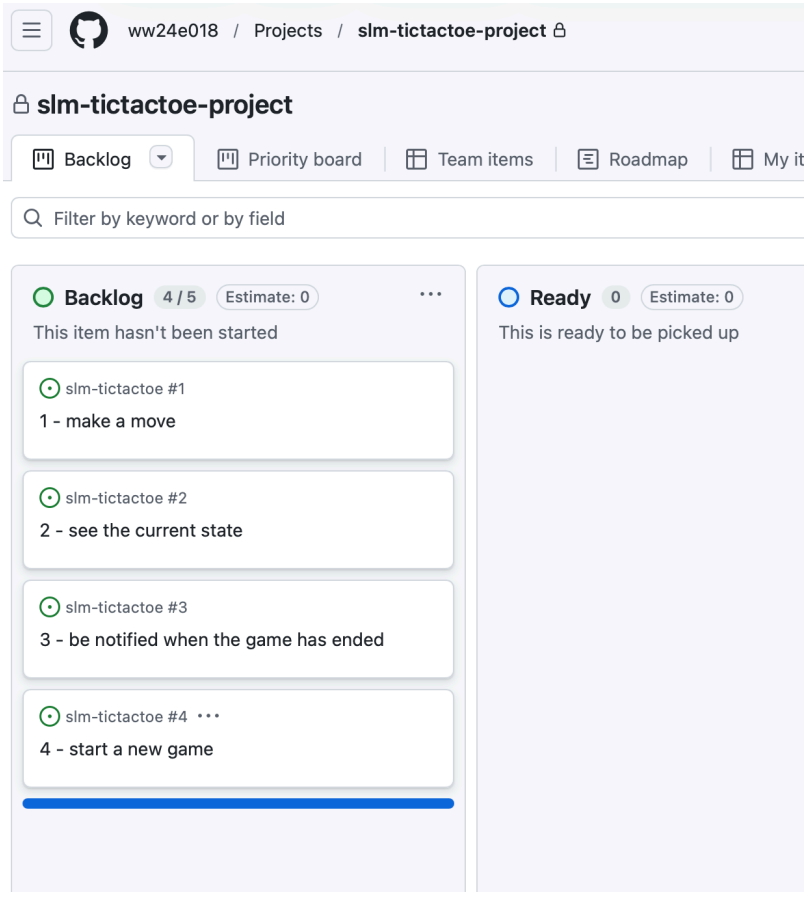


Screenshots of 2 and 4 and the final state:



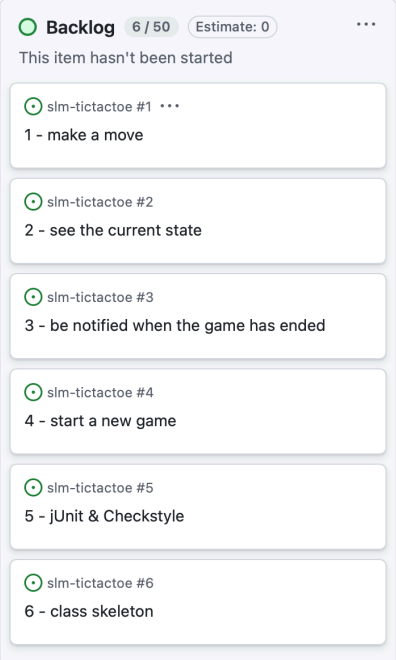


The board:

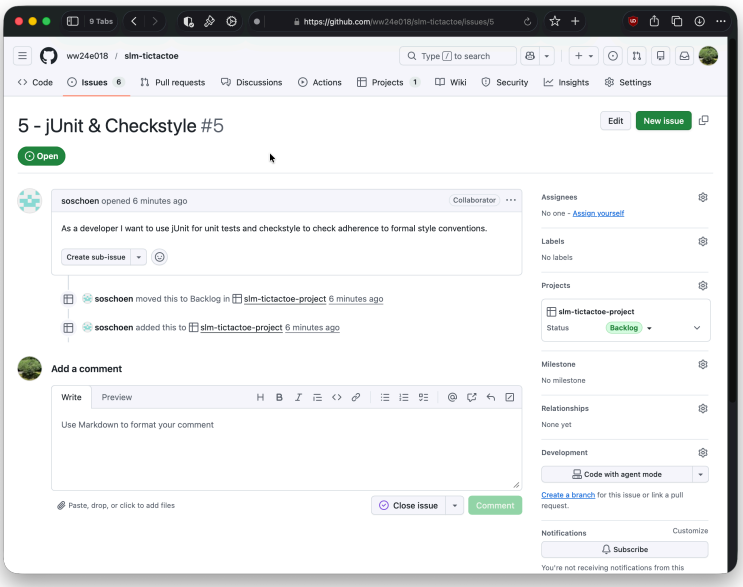

ww24e018 / Projects / slm-tictactoe-project
slm-tictactoe-project
Backlog
Priority board
Team items
Roadmap
My it
Filter by keyword or by field
Backlog 4 / 5 Estimate: 0
This item hasn't been started
slm-tictactoe #1
1 - make a move
slm-tictactoe #2
2 - see the current state
slm-tictactoe #3
3 - be notified when the game has ended
slm-tictactoe #4
4 - start a new game
Ready 0 Estimate: 0
This is ready to be picked up

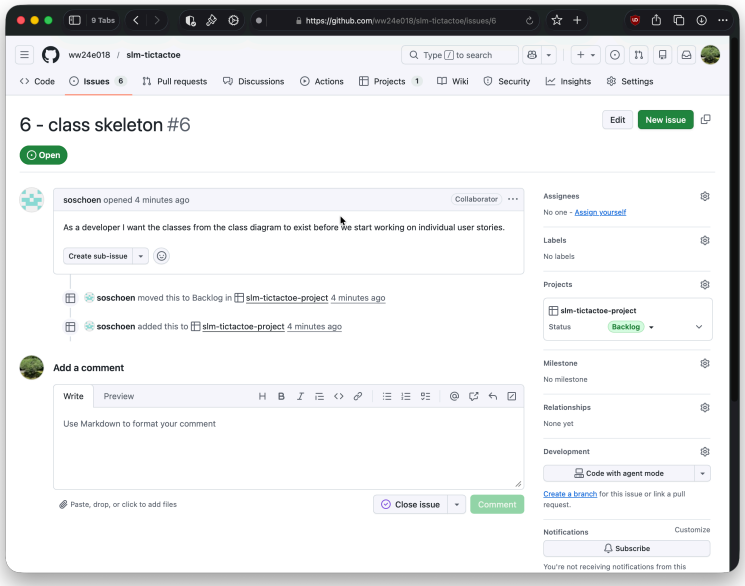
004 - Add additional issues to the kanban board

Additional issues were created to enable branch-naming for setup-tasks.
It was discovered there is a default WIP-Limit for the backlog column. It was at first randomly changed to 50 and later cleared altogether as we did not feel a necessity for a WIP limit in the backlog section in the context of this project.

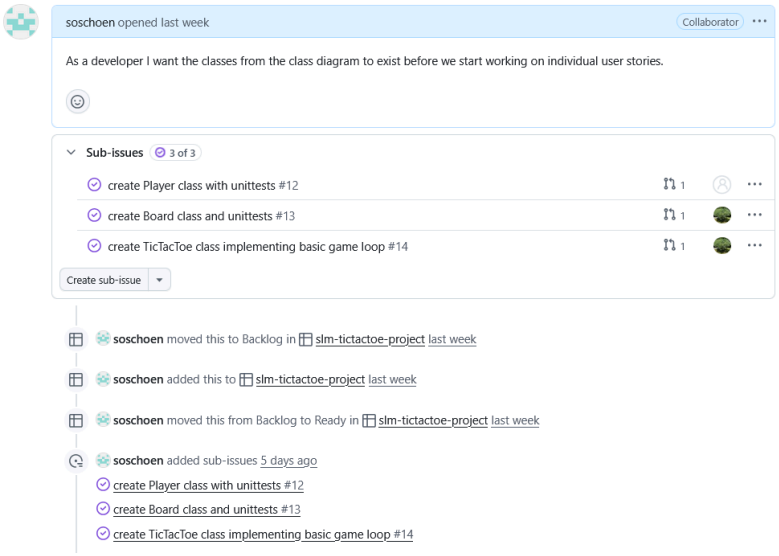


The following items were added:





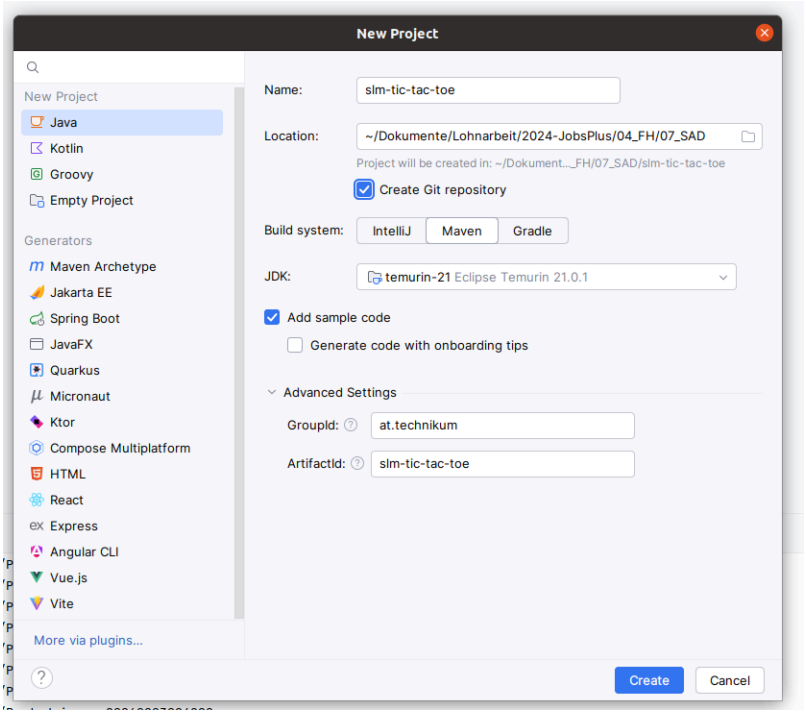
Issue #6 was further divided into 3 subissues:



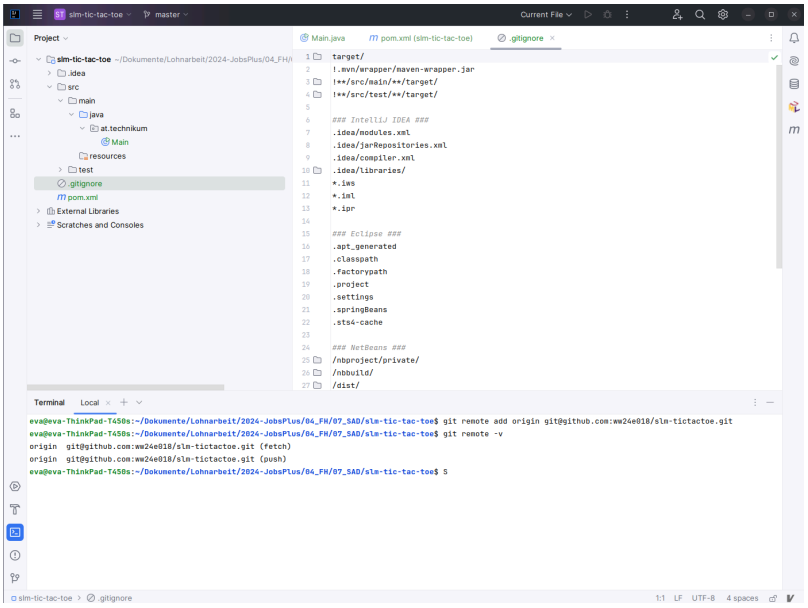
Additionally, a couple of new issues were created regarding workflows and ci/cd, in order to set everything up correctly before starting to work on the actual user stories.

005 - Create a new Java Maven project

A new Java Maven project was created in the IDE and a new git repository was initialised in the process.



006 - Set the GitHub repository as a remote repository



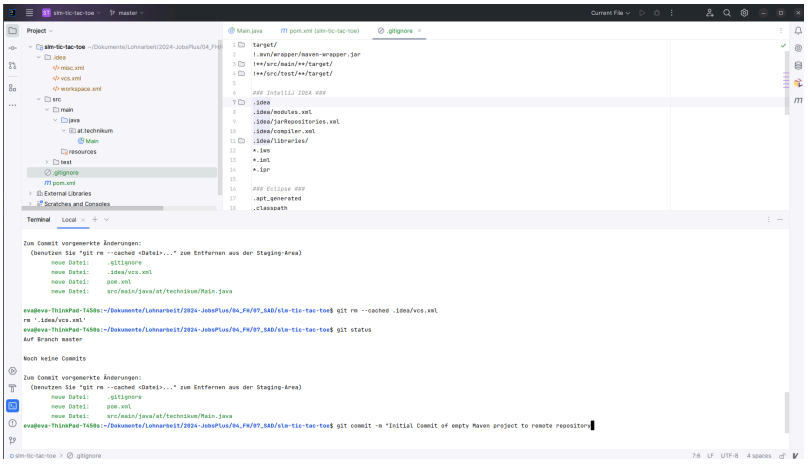
007 - Commit and push your blank Java project to the main branch

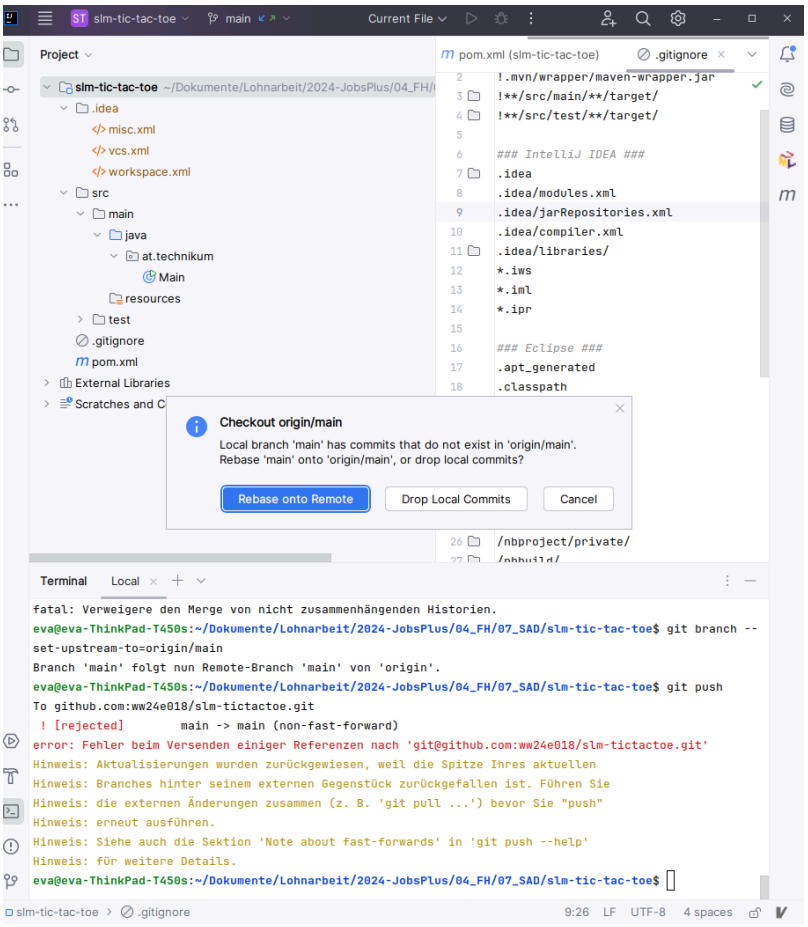
On our initial push to the main branch, we had a conflict between our local and remote branches.
Following differences lead to this:

1. Remote branch was named `main` and local branch `master`.
2. On creation of the repository in github we added a .gitignore which was different to the .gitignore file that was automatically generated by IntelliJ.

We resolved this by (see screenshot of history)

1. Rename master (local) to main (`git branch -m master main`)
2. Pull (`git pull git@github.com:ww24e018/slm-tictactoe.git main`)
3. Set upstream of local main to origin/main (`git branch --set-upstream-to=origin/main`)
4. Git push
5. Rebase & resolve conflict
6. git push



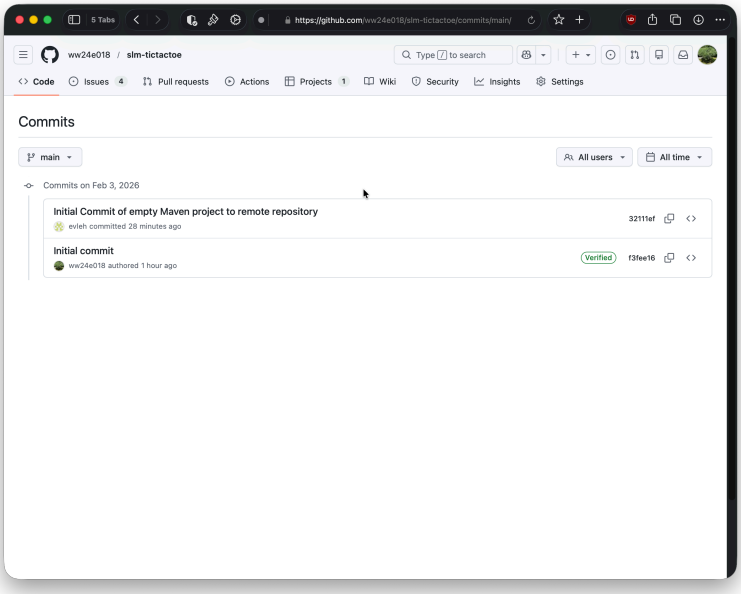


History leading to rebase button:

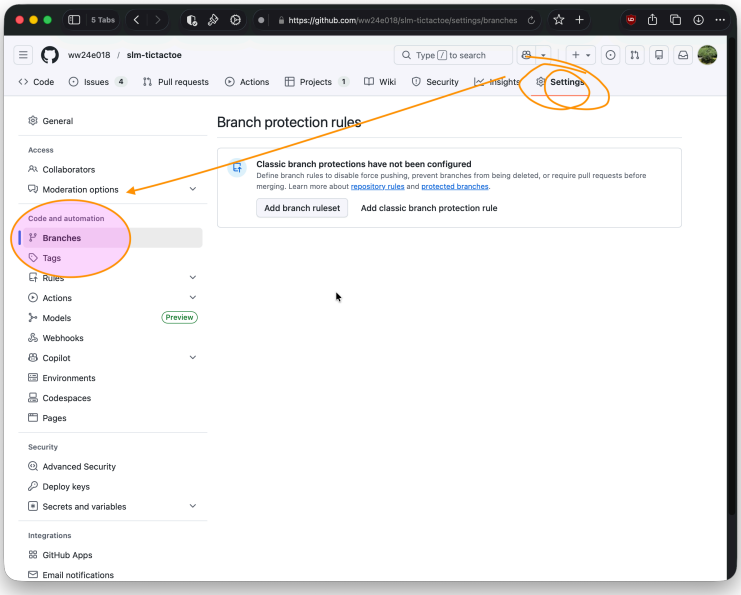
```
eva@eva-ThinkPad-T450s:~/Dokumente/Lohnarbeit/2024-JobsPlus/04_FH/07_SAD/slm-tic-tac-toe$ git push
fatal: Der aktuelle Branch master hat keinen Upstream-Branch.
Um den aktuellen Branch zu versenden und den Remote-Branch
als Upstream-Branch zu setzen, benutzen Sie

    git push --set-upstream origin master

eva@eva-ThinkPad-T450s:~/Dokumente/Lohnarbeit/2024-JobsPlus/04_FH/07_SAD/slm-tic-tac-toe$ git merge master
fatal: Verweigere den Merge von nicht zusammenhängenden Historien.
eva@eva-ThinkPad-T450s:~/Dokumente/Lohnarbeit/2024-JobsPlus/04_FH/07_SAD/slm-tic-tac-toe$ git branch -m master main
fatal: Branch 'main' existiert bereits.
eva@eva-ThinkPad-T450s:~/Dokumente/Lohnarbeit/2024-JobsPlus/04_FH/07_SAD/slm-tic-tac-toe$ git branch -m master main
eva@eva-ThinkPad-T450s:~/Dokumente/Lohnarbeit/2024-JobsPlus/04_FH/07_SAD/slm-tic-tac-toe$ git push
fatal: Der aktuelle Branch main hat keinen Upstream-Branch.
Um den aktuellen Branch zu versenden und den Remote-Branch
als Upstream-Branch zu setzen, benutzen Sie

    git push --set-upstream origin main

eva@eva-ThinkPad-T450s:~/Dokumente/Lohnarbeit/2024-JobsPlus/04_FH/07_SAD/slm-tic-tac-toe$ git push --set-upstream origin main
To github.com:ww24e018/slm-tictactoe.git
 ! [rejected]        main -> main (non-fast-forward)
error: Fehler beim Versenden einiger Referenzen nach 'git@github.com:ww24e018/slm-tictactoe.git'
Hinweis: Aktualisierungen wurden zurückgewiesen, weil die Spitze Ihres aktuellen
Hinweis: Branches hinter seinem externen Gegenstück zurückgefallen ist. Führen Sie
Hinweis: die externen Änderungen zusammen (z. B. 'git pull ...') bevor Sie "push"
Hinweis: erneut ausführen.
Hinweis: Siehe auch die Sektion 'Note about fast-forwards' in 'git push --help'
Hinweis: für weitere Details.
```

```
eva@eva-ThinkPad-T450s:~/Dokumente/Lohnarbeit/2024-JobsPlus/04_FH/07_SAD/slm-tic-tac-toe$ git pull
Es gibt keine Tracking-Informationen für den aktuellen Branch.
Bitte geben Sie den Branch an, welchen Sie zusammenführen möchten.
Siehe git-pull(1) für weitere Details.

    git pull <Remote-Repository> <Branch>

Wenn Sie Tracking-Informationen für diesen Branch setzen möchten, können Sie
dies tun mit:

    git branch --set-upstream-to=origin/<Branch> main

eva@eva-ThinkPad-T450s:~/Dokumente/Lohnarbeit/2024-JobsPlus/04_FH/07_SAD/slm-tic-tac-toe$ git pull git@github.com:ww24e018/slm-tictactoe.git main
Von github.com:ww24e018/slm-tictactoe
 * branch            main       -> FETCH_HEAD
fatal: Verweigere den Merge von nicht zusammenhängenden Historien.
eva@eva-ThinkPad-T450s:~/Dokumente/Lohnarbeit/2024-JobsPlus/04_FH/07_SAD/slm-tic-tac-toe$ git branch --set-upstream-to=origin/main
Branch 'main' folgt nun Remote-Branch 'main' von 'origin'.
eva@eva-ThinkPad-T450s:~/Dokumente/Lohnarbeit/2024-JobsPlus/04_FH/07_SAD/slm-tic-tac-toe$ git push
To github.com:ww24e018/slm-tictactoe.git
 ! [rejected]        main -> main (non-fast-forward)
error: Fehler beim Versenden einiger Referenzen nach 'git@github.com:ww24e018/slm-tictactoe.git'
Hinweis: Aktualisierungen wurden zurückgewiesen, weil die Spitze Ihres aktuellen
Hinweis: Branches hinter seinem externen Gegenstück zurückgefallen ist. Führen Sie
Hinweis: die externen Änderungen zusammen (z. B. 'git pull ...') bevor Sie "push"
Hinweis: erneut ausführen.
Hinweis: Siehe auch die Sektion 'Note about fast-forwards' in 'git push --help'
Hinweis: für weitere Details.
eva@eva-ThinkPad-T450s:~/Dokumente/Lohnarbeit/2024-JobsPlus/04_FH/07_SAD/slm-tic-tac-toe$ git push
Objekte aufzählen: 12, fertig.
Zähle Objekte: 100% (12/12), fertig.
Delta-Kompression verwendet bis zu 4 Threads.
Komprimiere Objekte: 100% (5/5), fertig.
Schreibe Objekte: 100% (10/10), 1.28 KiB | 657.00 KiB/s, fertig.
Gesamt 10 (Delta 0), Wiederverwendet 0 (Delta 0)
To github.com:ww24e018/slm-tictactoe.git
```

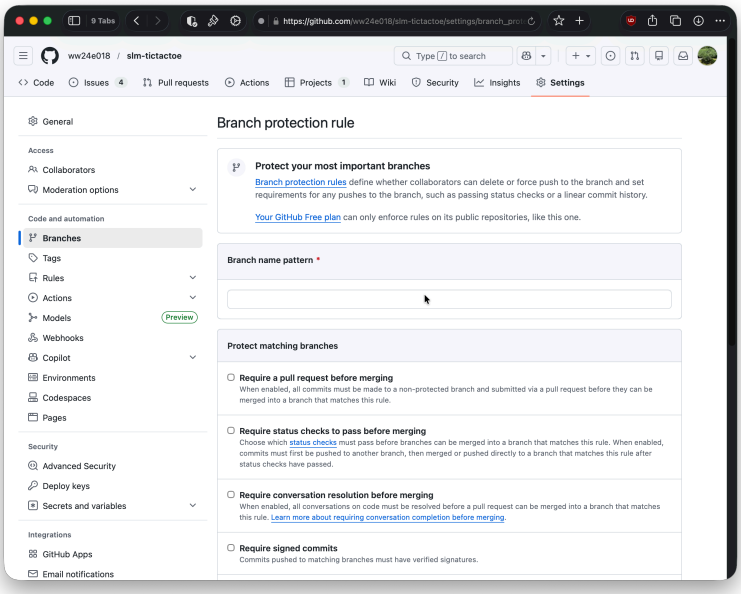
Screenshot of the commit history on GitHub:

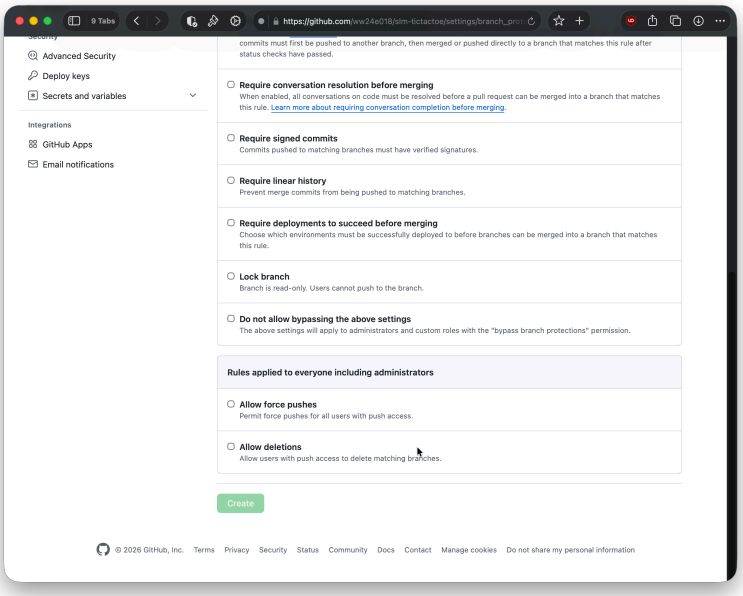


008 - Protect the main branch

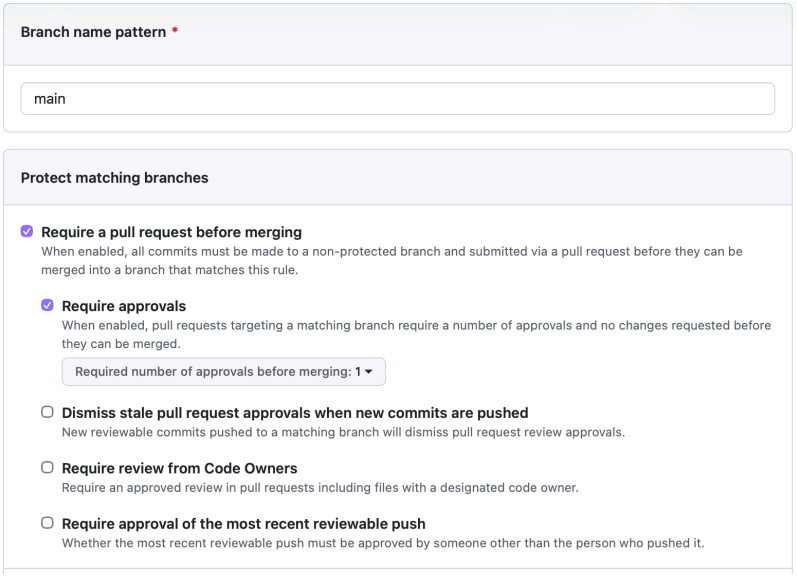


After reading documention about branch protection rules (repository rules / protected branches), it was decided to try the "add classic button":



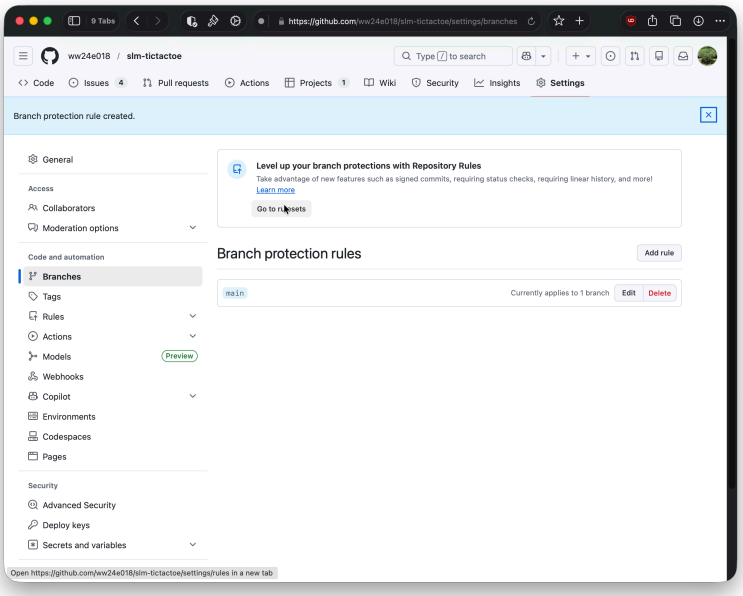


Defaults appearing after clicking the first checkbox



Other options were left at default.

After clicking 'create':


https://github.com/ww24e018/slm-tictactoe/settings/branches
ww24e018 / slm-tictactoe
Type / to search
Code
Issues 4
Pull requests
Actions
Projects 1
Wiki
Security
Insights
Settings
Branch protection rule created.
General
Access
Collaborators
Moderation options
Code and automation
Branches
Tags
Rules
Actions
Models
Preview
Webhooks
Copilot
Environments
Codespaces
Pages
Security
Advanced Security
Deploy keys
Secrets and variables
Level up your branch protections with Repository Rules
Take advantage of new features such as signed commits, requiring status checks, requiring linear history, and more!
Learn more
Go to rulesets
Branch protection rules
Add rule
main
Currently applies to 1 branch
Edit
Delete
Open https://github.com/ww24e018/slm-tictactoe/settings/rules in a new tab

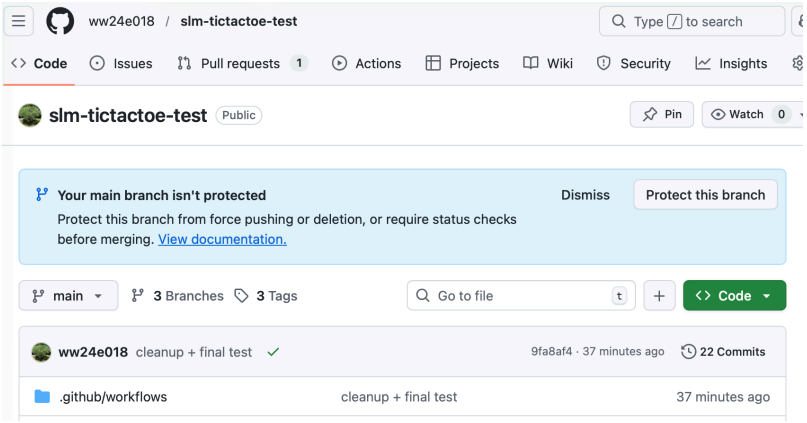
009 - CI_CD Scripts and workflows

The process of research and development

To simplify research and development - and to keep the submission repo free of commits doing trial and error on syntax and meaning - a decision was made to use a custom test-repository on GitHub to experiment more freely.

This repository is also public and is available here if a deeper view into that process is required (or desired).

It does not have the full spectrum of main-branch protections because its purpose is to serve as a playground for quick test runs that might require changes to be present on main.



The process of implementing results

was the usual of creating tickets/issues/board-items (! not userstories) on the board, creating branches, pushing branches, creating merge requests, requesting review, doing review, incorporating review and (at some point) merging.

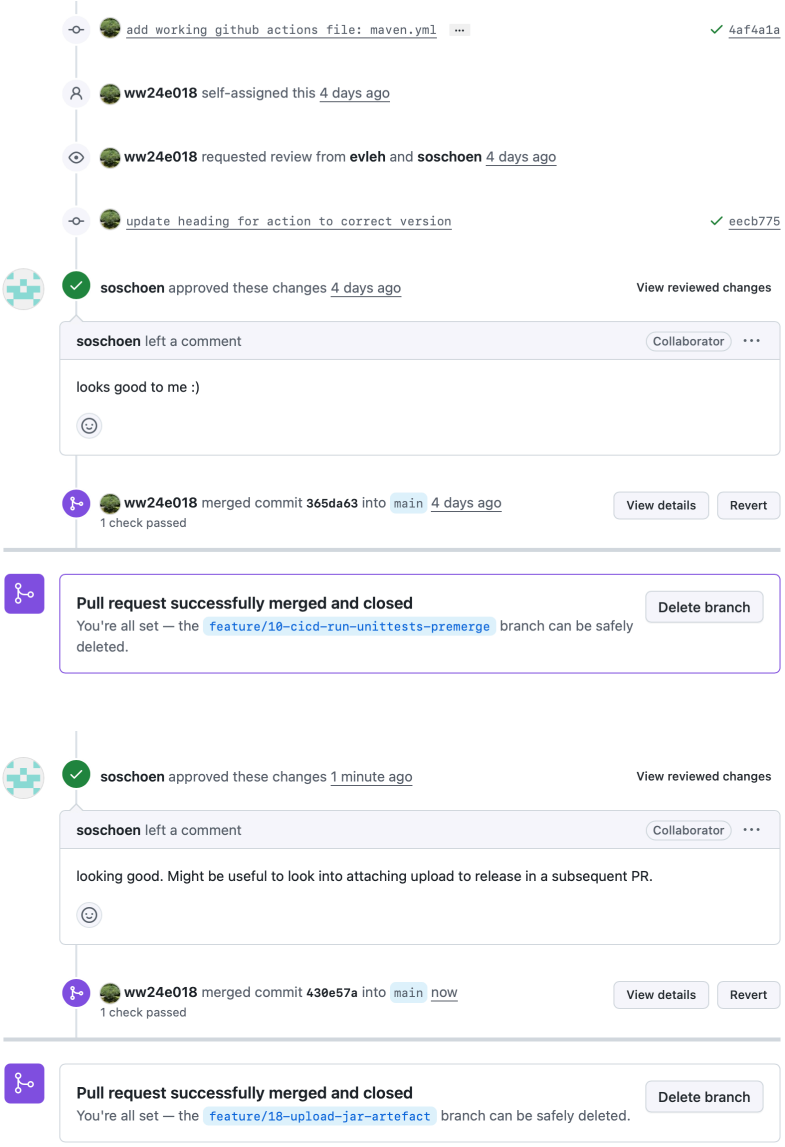
Links to the main 3 topical MRs/PRs:

- https://github.com/ww24e018/slm-tictactoe/pull/11
- https://github.com/ww24e018/slm-tictactoe/pull/19
- https://github.com/ww24e018/slm-tictactoe/pull/21

And links to the issues these close:

- https://github.com/ww24e018/slm-tictactoe/issues/10
- https://github.com/ww24e018/slm-tictactoe/issues/18
- https://github.com/ww24e018/slm-tictactoe/issues/20

Exemplary screenshot:



The suggestion made in the comment above lead to https://github.com/ww24e018/slm-tictactoe/issues/20 / https://github.com/ww24e018/slm-tictactoe/pull/21 .

Technical description

The process resulted in 2 yaml files:

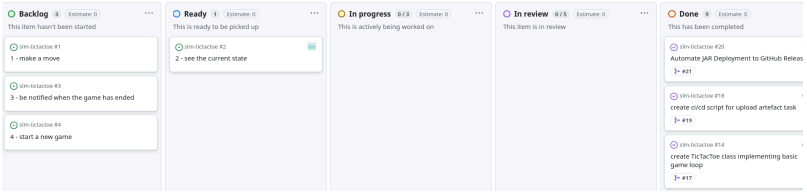
- `.github/workflows/maven.yml`
- `.github/workflows/maven-publish.yml`
  These names are the default ones given by the github-actions wizard. Later the question turned up if those two (who have an identical build-command) could be unified.

It was assumed this would be possible but would/might require research into and testing of more advanced github-actions-syntax to distinguish between the different trigger-events.

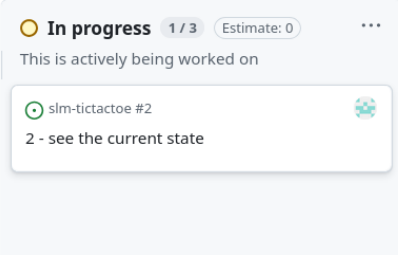
For now - while possibly slightly inefficient - they satisfy requirements.

010 - Working on User Stories

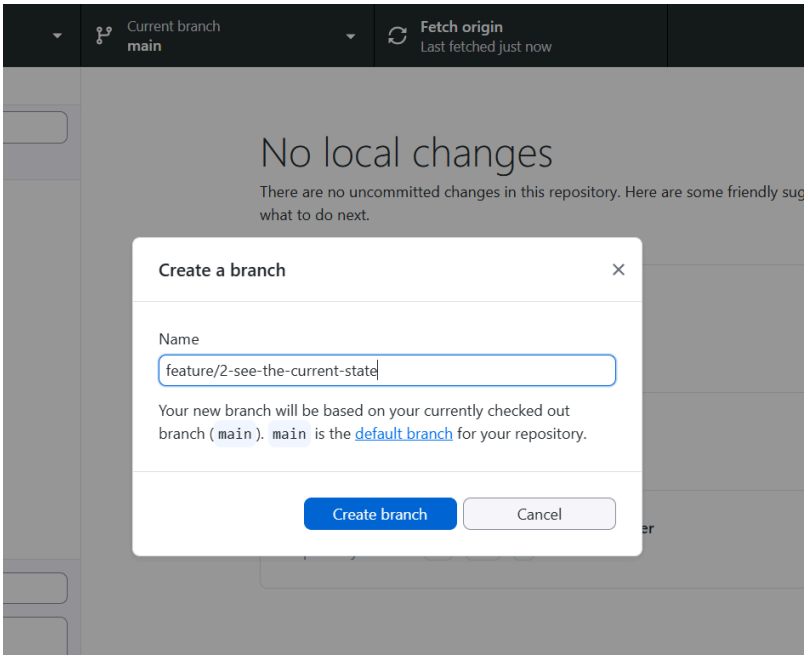
1. Once a user story is moved to `ready`, it is assigned to one or more contributors after checking in with each other in person.



2. When contributor starts actively working on the story, they move it to `In progress`.

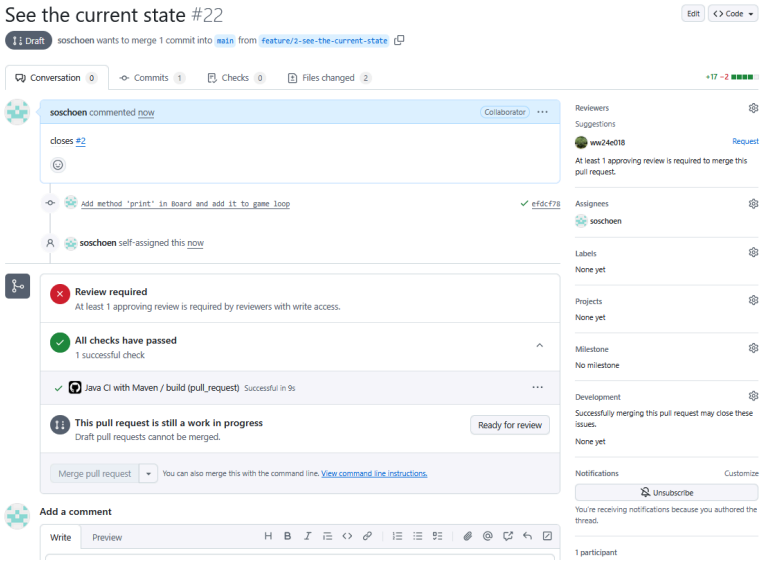


3. A new branch is created (locally) and published to remote.

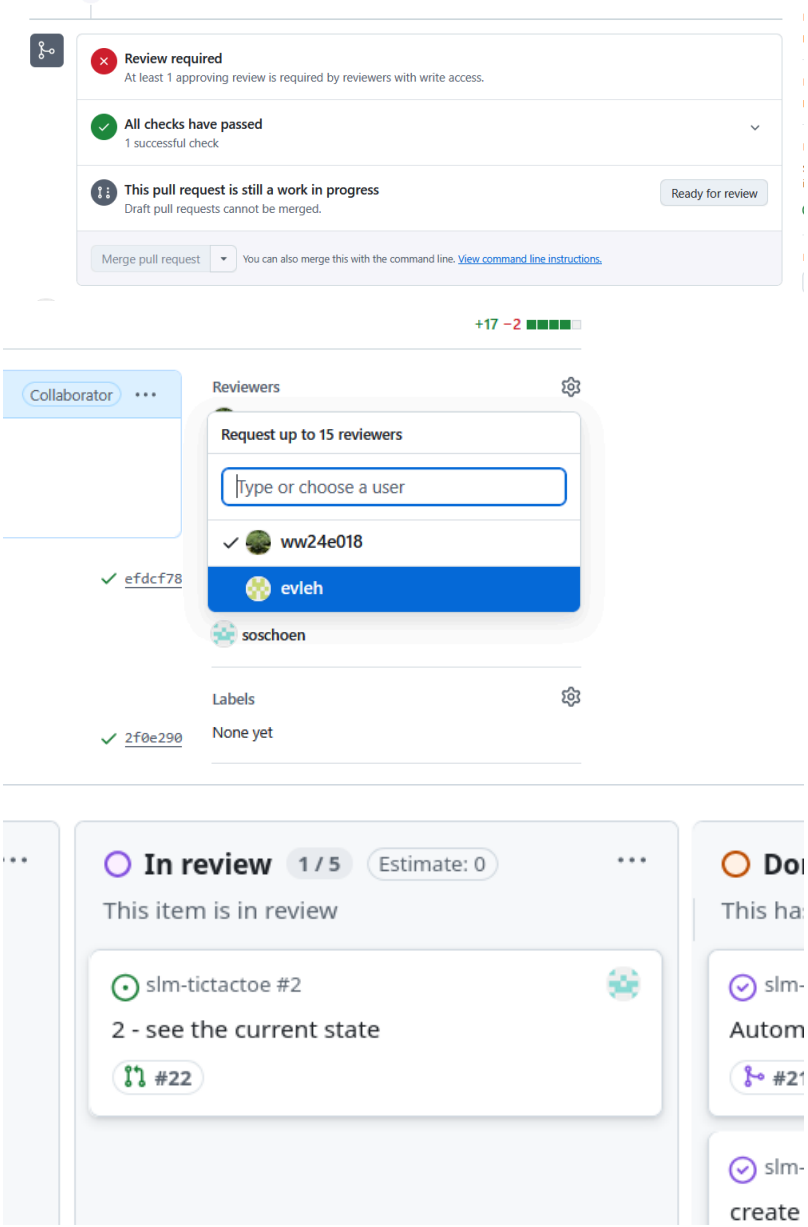


4. After the first commit ist ready, we push to remote and create a pull request which we mark as draft. This is to further visualize our work in progress to the rest of the team.

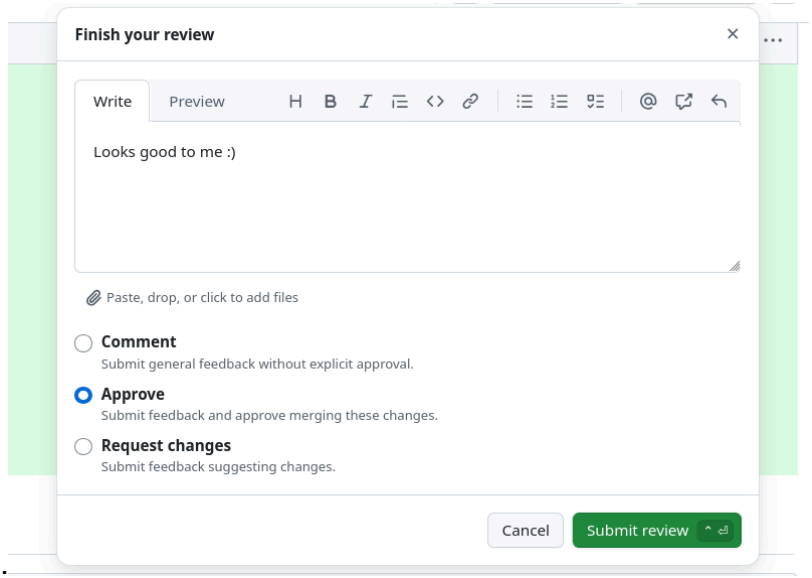
5. In our description we link the pull request to the issue.



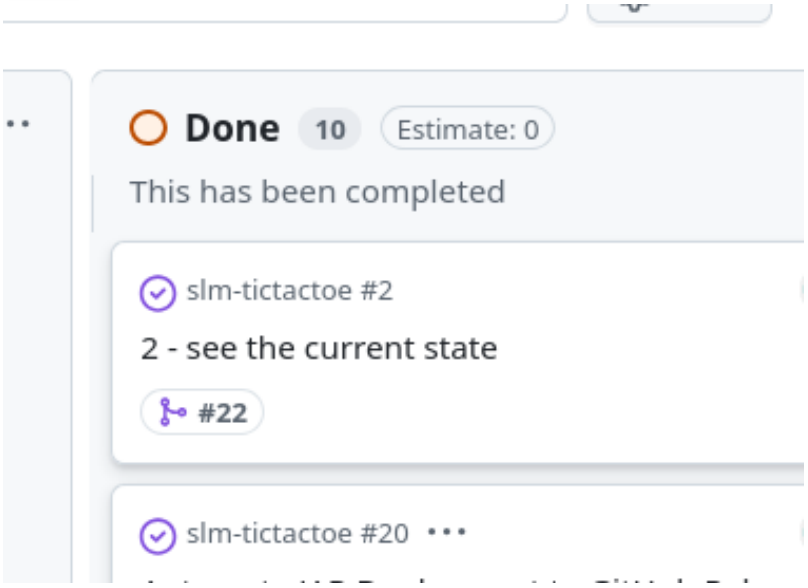
6. When the developer is done with the ticket and all checks have passed, they set the pull request to "ready for review", request a review and move the ticket to `In review`.



7. The reviewer writes a review. If there are no necessary changes, they can directly approve the request.



8. After the pull request is approved, the developer merges to main and moves the ticket to `Done`.



9. Sometimes the reviewer has questions, remarks, recommendations or corrections. These are posted as comments. If necessary or useful, they are also discussed in person. The review is then submitted as requesting changes.
10. The author makes amends to the code and answers to the comments left by the reviewer. After changes have been made, they re-request a review.
11. The reviewer who requested the changes reviews the author's answers and changes to the code and ultimately resolves the conversation in the comments.

12. Once they are satisfied with the result, they approve the pull request. Subsequently the feature branch is merged by the author (see steps 7-8).

011 - A final overview

A clean Kanban board - all issues have been moved to `done`:

Exemplary game run - everything works as it should:

Our big release: