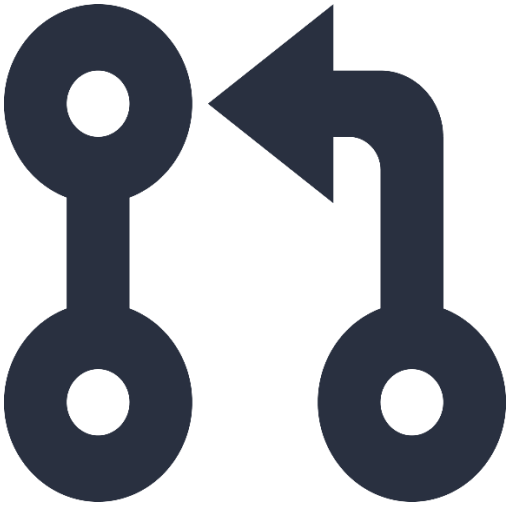
324-06B Pull Requests

Funktionsweise von Pull-Requests/Merge-Request

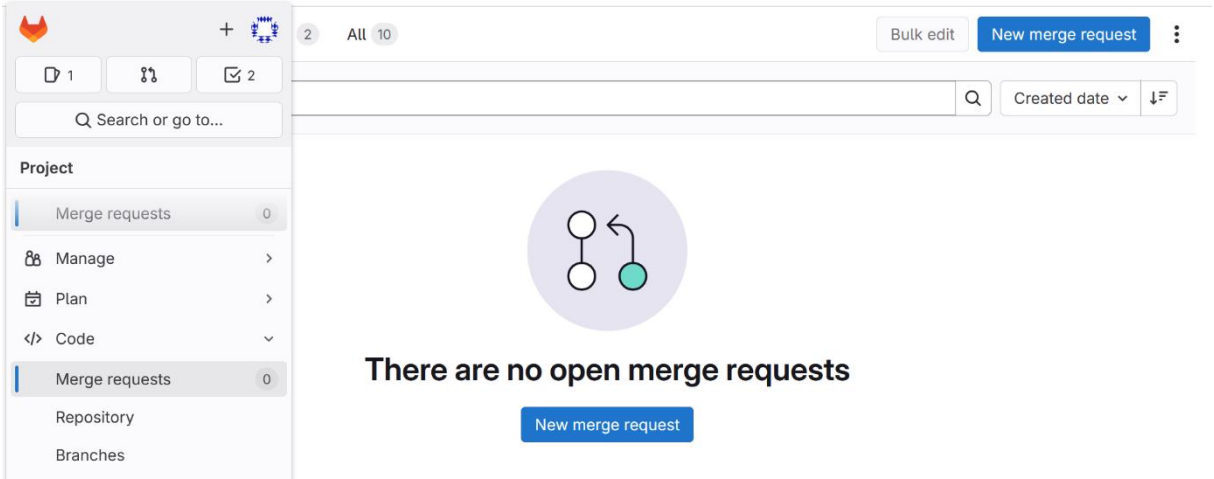


Pull-Rquests/Merge-Requests ermöglicht es Entwicklern, ihre Arbeit von einem separaten Entwicklungsbranch in den Hauptbranch des Projekts zu überführen.

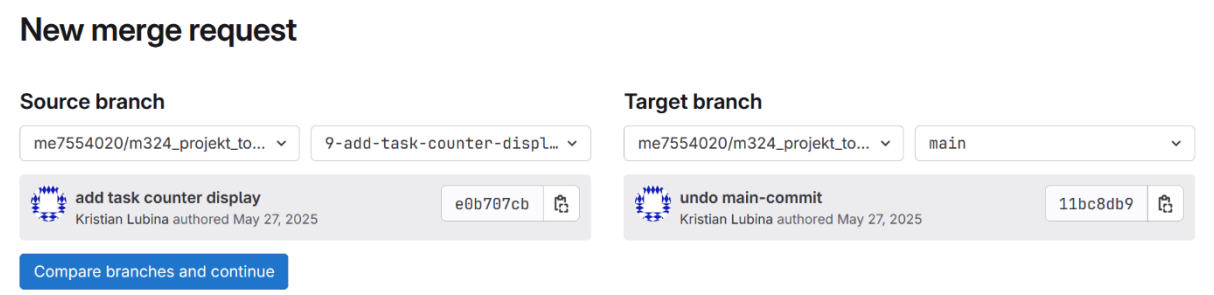
Ein wesentlicher Vorteil dieses Workflows liegt im integrierten Code-Review-Prozess. Andere Entwickler können die vorgeschlagenen Änderungen begutachten, Kommentare hinterlassen und Verbesserungsvorschläge einbringen, bevor der Code in den Hauptbranch integriert wird.

Pull-Request/Merge-Request in GitLab

Um in GitLab einen Merge-Request zu eröffnen geht man zur Merge-Request-Seite die in der Navigation von GitLab auffindbar ist.

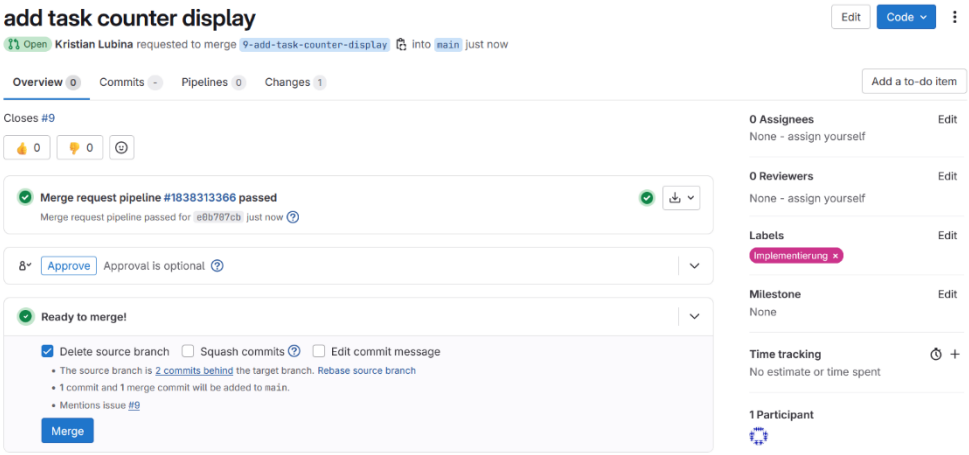


Nach dem man einen Merge-Request eröffnet hat kann man nun auswählen welche Branches man mergen möchte.

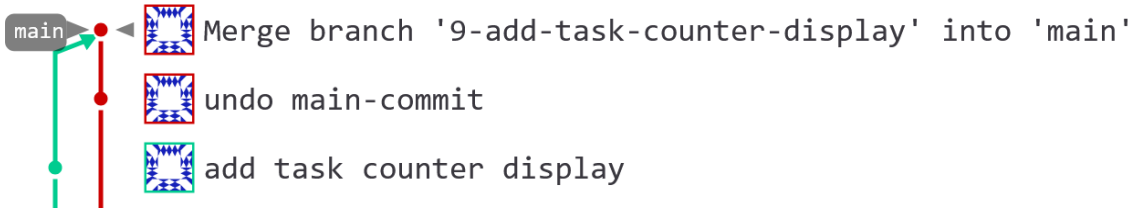


Falls man im Projekt eine Pipeline eingerichtet hat, so wie in unserem Fall, muss man zuerst einmal abwarten ob die Änderung die Build-Überprüfung bestehen. Wenn es keine Überprüfung allgemein gibt, dann kann man gerade direkt die zwei Branches mergen in dem man lediglich auf den merge-Button klickt.

Ebenfalls kann auch beim Mergen direkt bereits den Ursprungs-Branch löschen in dem man die Option auswählt, so wie es bei dem Beispiel im Screenshot es ist.



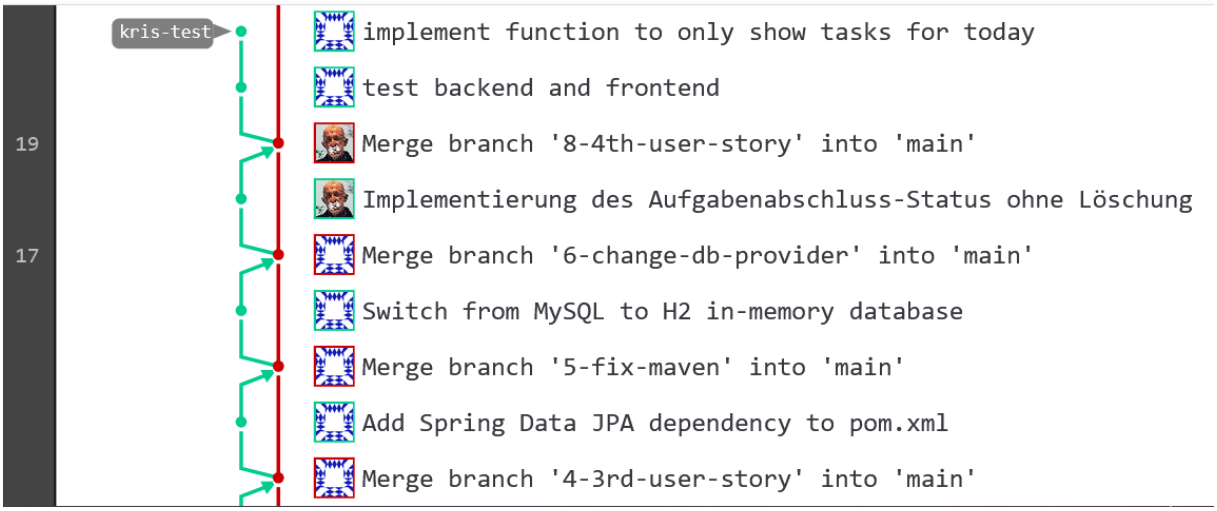
Danach werden die Änderungen des einen Branches in den anderen Branch verschmolzen. Das kann man auch in GitLab in der Navigation bei Code -> Repository Graph einsehen. Das Resultat ist eine Merge-Visualiserung wie aus dem Kapitel „Funktionsweise von Pull-Requests/Merge-Request“ dieser Dokumentation.



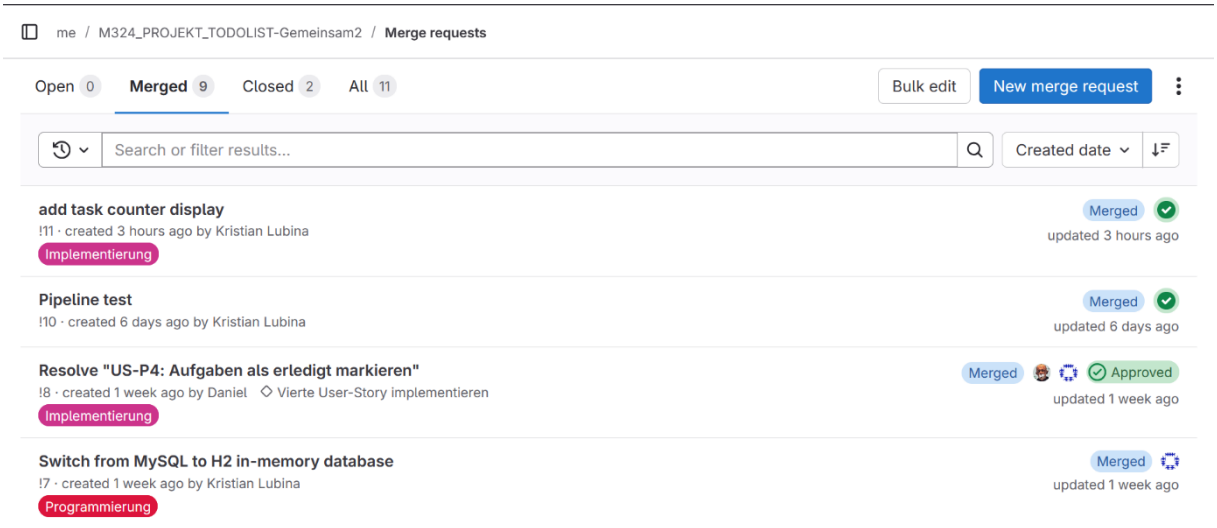
Pull-Request/Merge-Request in unserem Projekt

Bisherige Pull-Requests/Merge-Requests

Wir haben in diesem Software-Projekt bereits etliche male Pull-Requests/Merge-Requests durchgeführt.

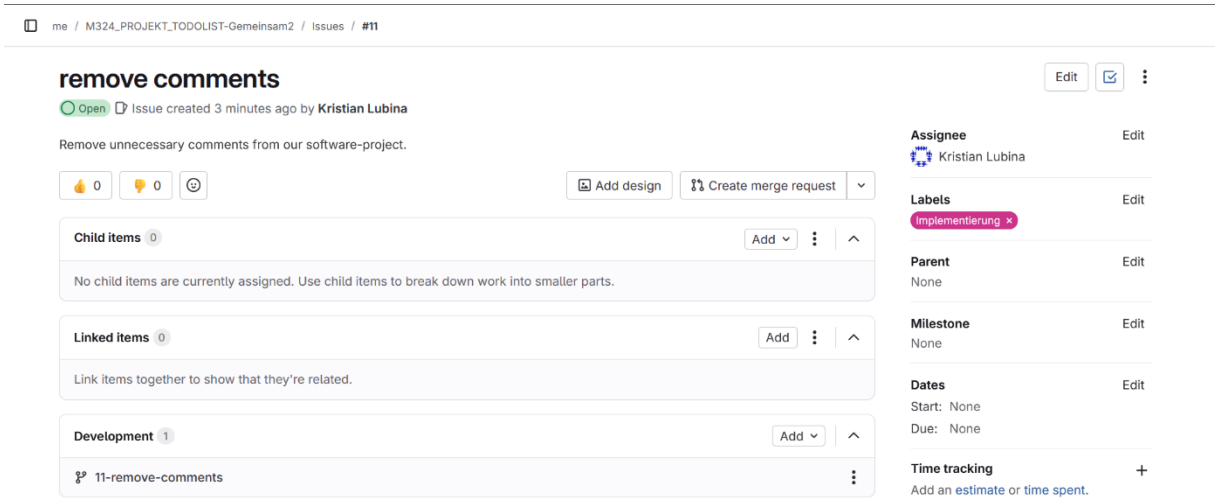


Genauer gesagt habe wir nach dem jetzigen Stand bereits neuen Merge-Requests durchgeführt.



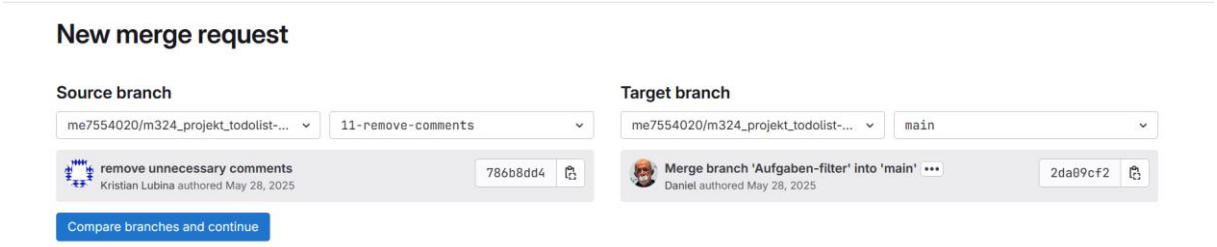
Weiterer demonstrativer Pull-Request/Merge-Request

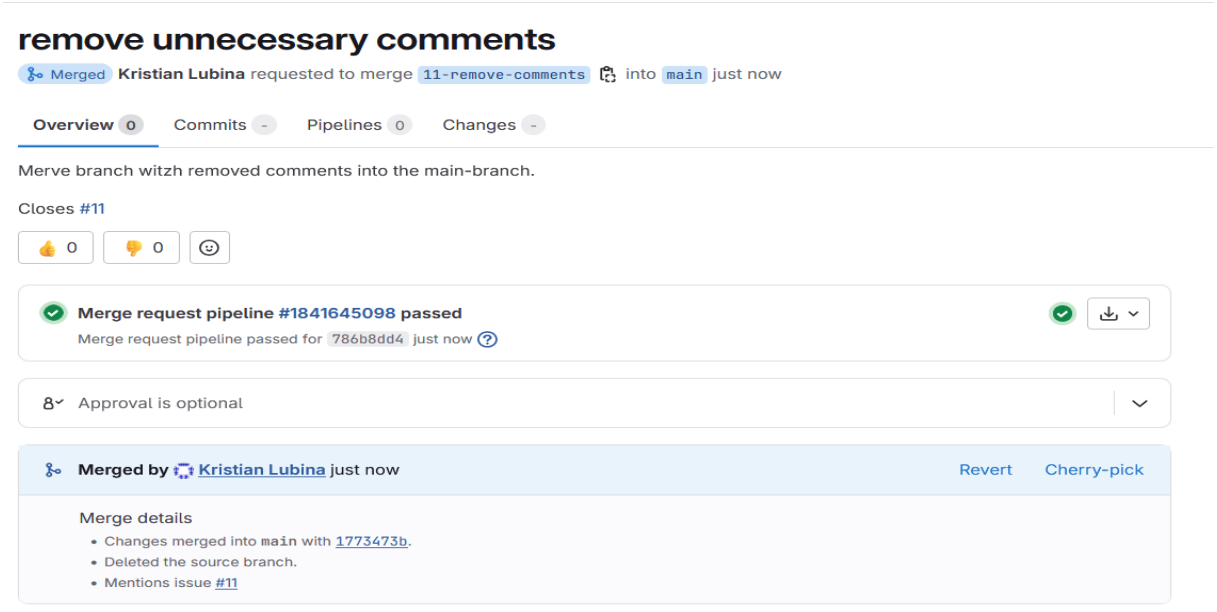
Nicht desto trotz werden wir noch einen Merge-Request durchführen für diese Sidequest. Zunächst einmal eröffnen wir ein Issue in dem wir das zu beiseitigte Problem definieren. Diesmal möchten wir überflüssige Kommentare im Code unseres Projektes entfernen.



Vom Issue her aus erstellen wir auch gerade den Feature-Branch „11-remove-comments“ in dem wir auch unsere Änderungen vornehmen werden.

Nach dem die Änderungen vorgenommen wurden und commited wurden, wird nun das Feature-Branch in den main-Branch gemerged.





Unsere Meinung zu Pull-Request/Merge-Request

Nutzen von Pull-Requests/Merge-Requests

Code-Review und Qualitätssicherung: Teammitglieder überprüfen Änderungen vor Integration, besonders wertvoll bei unserem Full-Stack-Projekt (React + Spring Boot).

Automatisierte Pipeline: Unsere .gitlab-ci.yml führt bei jedem Merge Request automatisch Backend-Build (Maven) und Frontend-Build (npm) aus, speichert Artefakte für 30 Tage.

Dokumentation: Nachvollziehbare Historie aller Änderungen, sichtbar bei unseren User Stories (Prioritäten, Fälligkeitsdatum, persistente Speicherung).

Wissenstransfer: Team lernt voneinander durch Code-Reviews über verschiedene Projektbereiche hinweg.

Erleichterung vs. Erschwerung und Einfluss auf Code-Qualität

Erleichterungen

- **Frühe Fehlererkennung**: CI-Pipeline verhindert defekten Code im Hauptbranch
- **Parallelentwicklung**: Verschiedene Features gleichzeitig in separaten Branches
- **Rollback-Möglichkeiten**: Problematische Änderungen leicht identifizierbar

Erschwerungen

- **Zeitaufwand**: Code-Reviews benötigen Zeit von Autor und Reviewern
- **Koordination**: Frontend- und Backend-Änderungen müssen abgestimmt werden
- **Verzögerungen**: Bei nicht verfügbaren Reviewern oder umfangreichen Änderungen

Positive Auswirkungen auf Code-Qualität

- **Build-Stabilität**: Automatisierte Builds für Maven-Backend und npm-Frontend
- **Architektur-Compliance**: Neue Features folgen bestehender Struktur
- **Datenkonsistenz**: Korrekte JPA-Annotationen bei persistenter Speicherung durch Reviews

Unsere endgültige Meinung

Trotz initialem Mehraufwand führen Pull-Requests zu höherer Code-Qualität, stabilerer Anwendung und reduzierter Fehlerrate. Somit sind Pull-Requests eine gute Sache.