



# Tag 1: Einführung in Git und GitLab

08.07.2024, Daniel Krämer

© Copyright 2024 anderScore GmbH



A large, semi-transparent watermark or background image is visible across the slide. It shows a modern multi-story building with many windows on the left, and on the right, a view of a church with two prominent towers and a tall tower with a flag on the far right under a clear blue sky.

**HECKER**  
CONSULTING

# Agenda

- **Tag 1 – Einführung in Git und GitLab**
  - Einführung & Kursüberblick
  - Grundlagen von Git
  - Git Rebase und Merge-Strategien
  - Git Remote
  - Grundlagen von GitLab
- **Tag 2 – Git-Workflows, CI/CD, GitLab CI**
  - Git-Workflow im Team
  - Gitflow-Workflow
  - Tags, Releases & deren Verwaltung
  - Einführung in GitLab CI/CD & gitlab-ci.yml
  - GitLab Runner
- **Tag 3 – Docker, GitOps, Deployment-Strategien**
  - Entwicklung mit Docker
  - Container/Docker-Registry
  - Erstellen von Release- und Tagged-Images
  - GitOps Grundlagen
  - Möglichkeiten des Deployments & Verwaltung von Konfiguration
  - Abschlussübung & Diskussion

# Agenda

- **Tag 1 – Einführung in Git und GitLab**
  - Einführung & Kursüberblick
  - Grundlagen von Git
  - Git Rebase und Merge-Strategien
  - Git Remote
  - Grundlagen von GitLab
- **Tag 2 – Git-Workflows, CI/CD, GitLab CI**
  - Git-Workflow im Team
  - Gitflow-Workflow
  - Tags, Releases & deren Verwaltung
  - Einführung in GitLab CI/CD & gitlab-ci.yml
  - GitLab Runner
- **Tag 3 – Docker, GitOps, Deployment-Strategien**
  - Entwicklung mit Docker
  - Container/Docker-Registry
  - Erstellen von Release- und Tagged-Images
  - GitOps Grundlagen
  - Möglichkeiten des Deployments & Verwaltung von Konfiguration
  - Abschlussübung & Diskussion

# Einführung in **GitLab**

## Inhalt

- Einführung
  - Was ist GitLab?
  - Git vs. GitLab
  - Versionen
- Live Demo
  - Projekte erstellen
  - Gruppen
  - Arbeiten im Projekt
  - Planing Tools
  - Branches und Merge Requests
- Übung

## Was ist GitLab?

- 2011 von Dimitri Saporoschez und Valery Sizov entwickelt
- Service zur Softwareentwicklung und Versionsverwaltung
- Hosting von Git Remote Repositories als Projekte
- Zusätzliche Funktionen:
  - Issue Tracking: Verwaltung/Verfolgung von Aufgaben
  - Dokumentation
  - CI/CD (Continuous Integration/Continuous Deployment)



## Git vs. GitLab

### Git

- Verteiltes Versionskontrollsystem zur Verwaltung von Dateien
- Verfolgung und Protokollierung von Änderungen

### GitLab

- Umfassende Entwicklungs- und DevOps-Plattform
- Erweitert Git um Tools und Funktionalitäten für Projektmanagement, Zusammenarbeit und Automatisierung
- Zentrale Plattform zur Verwaltung von Projekten

## Versionen

- Free
  - Kostenlos und Open Source
  - Konzipiert für persönliche Projekte
  - 400 compute minutes für CI/CD
  - Support nur über GitLab Forum
- Premium
  - Für mittlere bis große Teams
  - Fortgeschrittene Features im Bereich Projektmanagement, Code Reviews sowie CI/CD
  - 10.000 compute minutes für CI/CD
  - Support bei Problemen

- Ultimate
  - Für Unternehmen und große Organisationen
  - Umfangreiche Statistiken zur Analyse
  - Erweiterte Funktionen insbesondere im Bereich Security und Automatisierung
  - 50.000 compute minutes für CI/CD
- Verwendung als SaaS (GitLab.com) oder selbst gehostete Instanz im eigenen Netzwerk
- Zusätzliche Add-ons mit Fokus auf AI-Unterstützung für Premium und Ultimate

# GitLab – Einführung



GitLab

# Startseite und Projekte

# Startseite & Projekte

Your work / Projects / New project

## Create new project



### Create blank project

Create a blank project to store your files, plan your work, and collaborate on code, among other things.



### Create from template

Create a project pre-populated with the necessary files to get you started quickly.



### Import project

Migrate your data from an external source like GitHub, Bitbucket, or another instance of GitLab.



### Run CI/CD for external repository

Connect your external repository to GitLab CI/CD.

You can also create a project from the command line. [Show command](#)

**DEMO**

# Startseite & Projekte

Your work / Projects / New project / Create blank project



## Create blank project

Create a blank project to store your files, plan your work, and collaborate on code, among other things.

### Project name

My awesome project

Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, dashes, or spaces.

### Project URL

<https://gitlab.example.de/testuser/>

### Project slug

/ my-awesome-project

### Visibility Level

Private

Project access must be granted explicitly to each user. If this project is part of a group, access is granted to members of the group.

Internal

The project can be accessed by any logged in user except external users.

Public

The project can be accessed without any authentication.

### Project Configuration

Initialize repository with a README

Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.

Enable Static Application Security Testing (SAST)

Analyze your source code for known security vulnerabilities. [Learn more](#).

[Create project](#)

[Cancel](#)

**DEMO**

## Neues Projekt erstellen

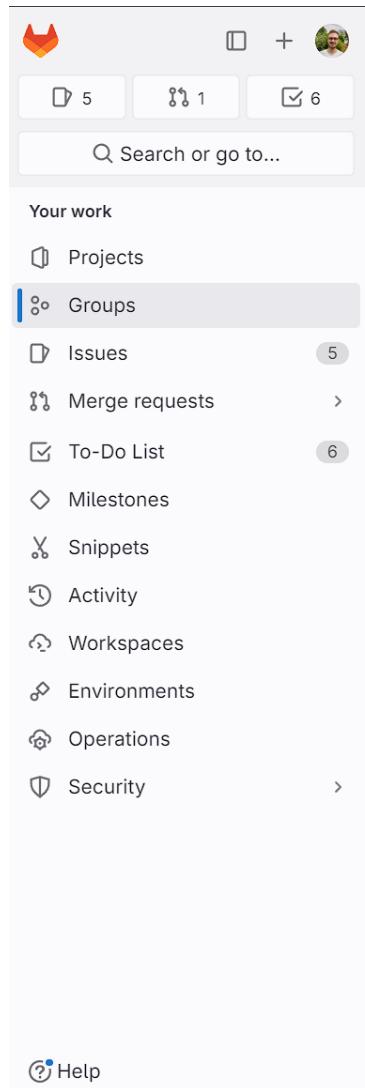
- URL
  - Project URL + Project Slug bestimmen die URL, unter der das neue Projekt aufgerufen werden kann
- Visibility Level
  - Gibt an, welche User Zugriff auf das Projekt erhalten
  - Internal als zusätzliche Option bei eigenen GitLab Instanzen für angemeldete User und Guest Accounts
- Project Configuration
  - Mit einer README wird eine erste Datei und ein initialer Commit angelegt
  - Sinnvoll wenn man zuerst das Remote Repository anlegt um direkt clonen zu können
  - Bei bestehendem lokalen Repository sollte man ein leeres Remote Repository anlegen

**DEMO**

GitLab

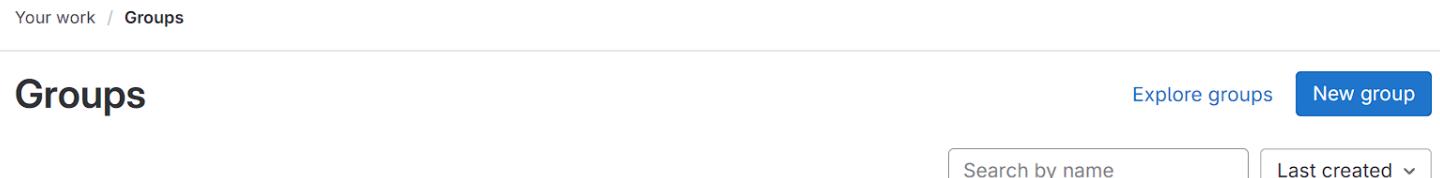
# Groups

# Groups



The sidebar navigation includes:

- Profile icon
- Dashboard icons: 5 projects, 1 merge request, 6 issues
- Search bar: Search or go to...
- Navigation links:
  - Projects
  - Groups** (selected)
  - Issues (5)
  - Merge requests
  - To-Do List (6)
  - Milestones
  - Snippets
  - Activity
  - Workspaces
  - Environments
  - Operations
  - Security (with dropdown arrow)
  - Help



The page shows the 'Groups' section with the following details:

- Header: Your work / Groups
- Title: Groups
- Buttons: Explore groups, New group
- Search bar: Search by name, Last created
- Groups list:
  - example-group (E) - Example group description (1 member)
  - Trainings (T) - Trainings (6 members)

**DEMO**

- Gruppen können zur Verwaltung von einem oder mehreren Projekten gleichzeitig verwendet werden
  - Projekte können innerhalb von Gruppen angelegt werden
  - Dadurch befinden diese sich im **Namespace** der Gruppe, bspw. [https://gitlab.example.de/my\\_group/my\\_project](https://gitlab.example.de/my_group/my_project)
  - Verwaltung von Rechten auf Gruppenebene
  - Mitglieder erhalten Zugriff auf alle Projekte innerhalb einer Gruppe
- Können zur Kommunikation verwendet werden
- Abrufen von gruppenspezifischen Statistiken und Aktivitäten
- Können in Untergruppen aufgeteilt werden
- Gruppeneinteilungen und Strukturen auf individuelle Bedürfnisse anpassbar

**DEMO**

# Groups

Your work / Groups / New group / Create group

## Create group



Groups allow you to manage and collaborate across multiple projects. Members of a group have access to all of its projects.

Groups can also be nested by creating subgroups.

### Group name

Must start with letter, digit, emoji, or underscore. Can also contain periods, dashes, spaces, and parentheses.



Your group name must not contain a period if you intend to use SCIM integration, as it can lead to errors.

### Group URL

### Visibility level

Who will be able to see this group? [View the documentation](#)

Private

The group and its projects can only be viewed by members.

Internal

The group and any internal projects can be viewed by any logged in user except external users.

Public

The group and any public projects can be viewed without any authentication.

### Now, personalize your GitLab experience

We'll use this to help surface the right features and information to you.

### Role

### Who will be using this group?

My company or team

Just me

### What will you use this group for?

### Invite Members (optional)

Invited users will be added with developer level permissions. [View the documentation](#) to see how to change this later.

### Email 1

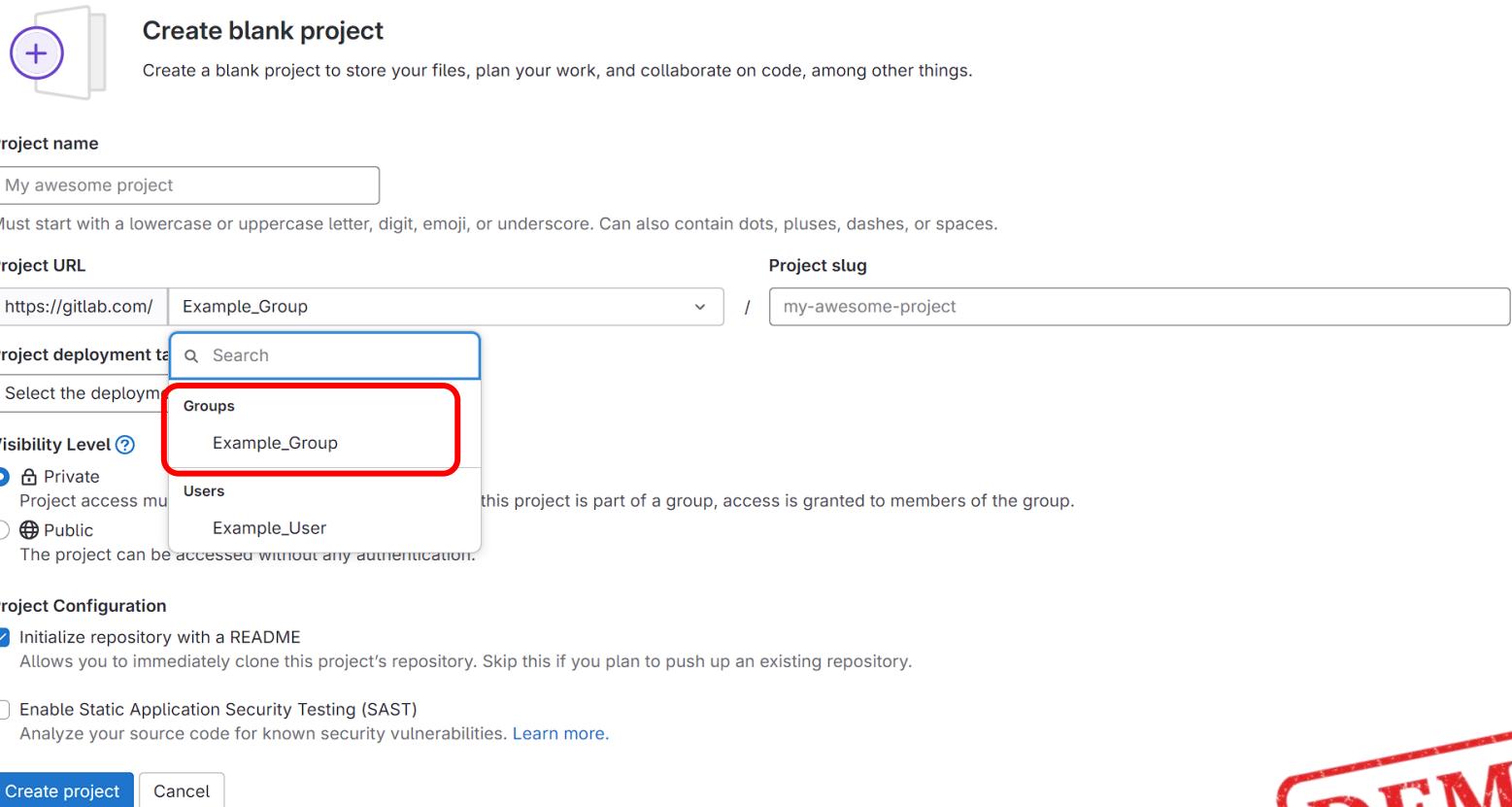
+ Invite another member

**DEMO**

# Groups

- Anlegen eines Projektes innerhalb einer Gruppe über Button „New Project“ auf Startseite/Gruppenseite

Your work / Projects / New project / Create blank project



Create blank project

Create a blank project to store your files, plan your work, and collaborate on code, among other things.

**Project name**

Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, dashes, or spaces.

**Project URL**

 **Project slug** / my-awesome-project

**Project deployment target**

**Select the deployment target**

**Groups**

**Example\_Group**

**Visibility Level**   Private  Public

Project access must be granted by a member of the group. this project is part of a group, access is granted to members of the group.

**Users**

**Example\_User** The project can be accessed without any authentication.

**Project Configuration**

Initialize repository with a README Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.

Enable Static Application Security Testing (SAST) Analyze your source code for known security vulnerabilities. [Learn more](#).

**Create project** **Cancel**

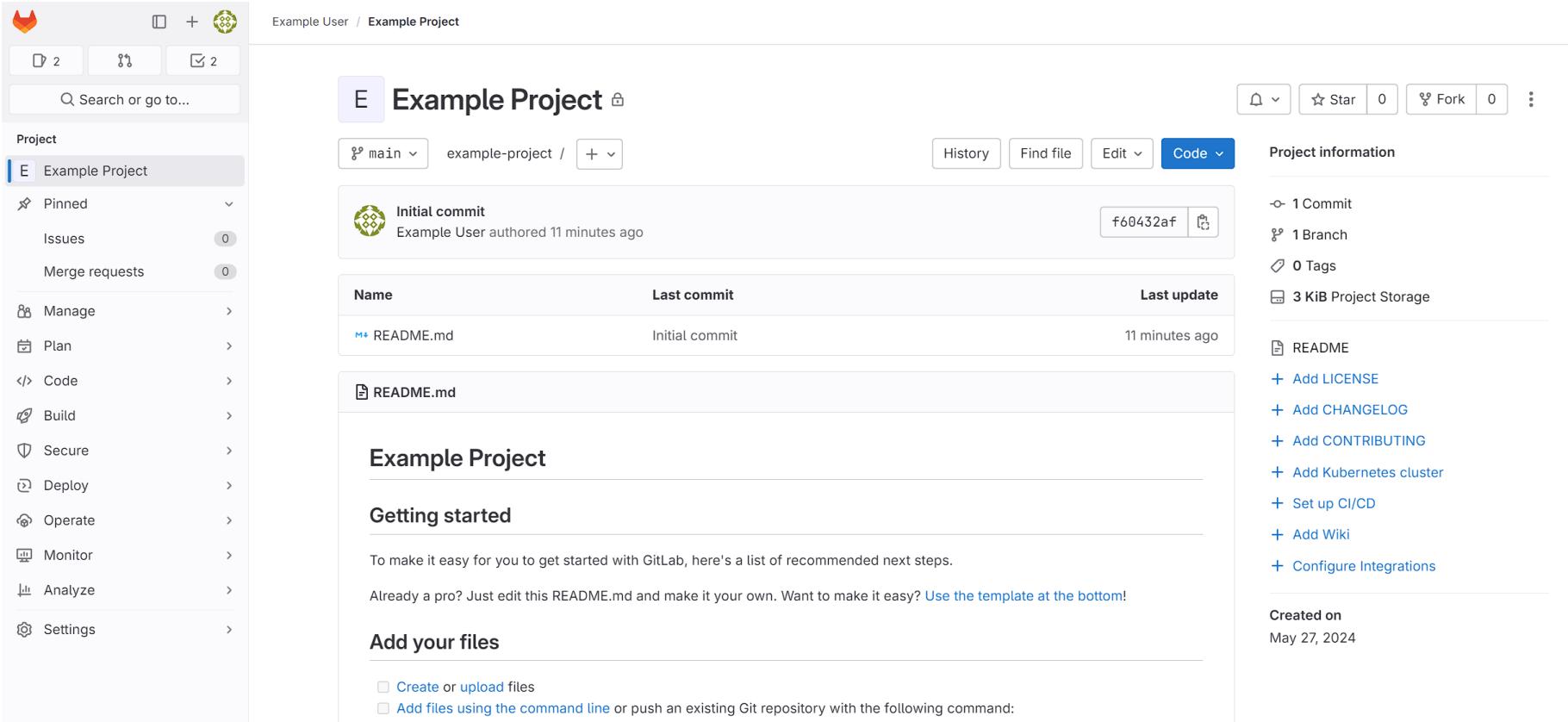
**DEMO**

GitLab

# Clonen von Projekten

**DEMO**

# Clonen von Projekten



The screenshot shows a GitLab project interface for 'Example Project'. The left sidebar includes pinned items, issues (0), merge requests (0), and various management sections like Plan, Code, Build, Secure, Deploy, Operate, Monitor, Analyze, and Settings. The main content area displays the project's README.md file, which contains the text 'Example Project', 'Getting started', and instructions for adding files. It also shows a single commit from 'Example User' with hash f60432af. The right sidebar provides project information: 1 Commit, 1 Branch, 0 Tags, 3 KiB Project Storage, and links to README, Add LICENSE, Add CHANGELOG, Add CONTRIBUTING, Add Kubernetes cluster, Set up CI/CD, Add Wiki, and Configure Integrations. A 'Created on' section indicates the project was created on May 27, 2024.

Project information

- 1 Commit
- 1 Branch
- 0 Tags
- 3 KiB Project Storage

README

+ Add LICENSE

+ Add CHANGELOG

+ Add CONTRIBUTING

+ Add Kubernetes cluster

+ Set up CI/CD

+ Add Wiki

+ Configure Integrations

Created on

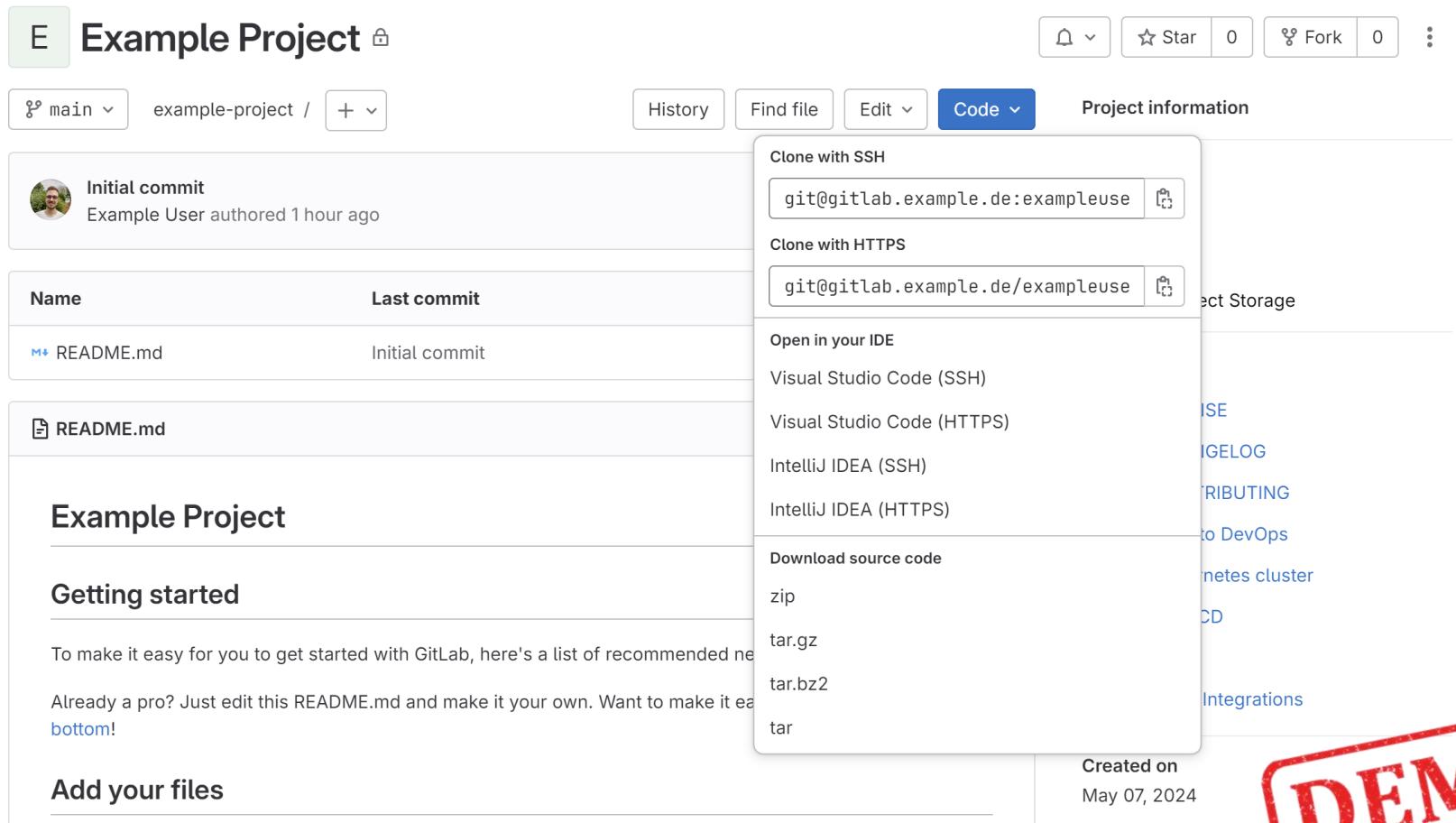
May 27, 2024

**DEMO**

# Clonen von Projekten

- In Projektübersicht **Code** auswählen

Example User / Example Project



The screenshot shows a GitLab project page for 'Example Project'. The 'Code' dropdown menu is open, displaying cloning options via SSH and HTTPS, as well as IDE integration and download formats.

**Code dropdown menu:**

- Clone with SSH: git@gitlab.example.de:exampleuse
- Clone with HTTPS: git@gitlab.example.de/exampleuse
- Open in your IDE
  - Visual Studio Code (SSH)
  - Visual Studio Code (HTTPS)
  - IntelliJ IDEA (SSH)
  - IntelliJ IDEA (HTTPS)
- Download source code
  - zip
  - tar.gz
  - tar.bz2
  - tar

**Project Information:**

- History
- Find file
- Edit
- Code
- Project information

**Project Details:**

- Initial commit by Example User authored 1 hour ago
- Name: README.md, Last commit: Initial commit
- File: README.md

**Getting started:**

To make it easy for you to get started with GitLab, here's a list of recommended ne

Already a pro? Just edit this README.md and make it your own. Want to make it ea bottom!

**Add your files:**

Created on May 07, 2024

**DEMO**

# Clonen von Projekten

- Clonen mittels SSH Key oder über HTTPS möglich
- Im gewünschten Zielverzeichnis den Befehl  
`git clone` in Kombination mit der kopierten URL ausführen
- Zum Clonen über SSH muss ein öffentlicher SSH Schlüssel hinterlegt werden

**DEMO**

## SSH Keys

- Können zur verschlüsselten Kommunikation zwischen lokalem Repository und GitLab Server verwendet werden
- Sinnvolle Alternative zur wiederholten Verwendung von Username und Passwort
- SSH Keys besteht aus einem Schlüsselpaar
  - Öffentlicher Schlüssel wird auf den GitLab Server hinterlegt
  - Privater Schlüssel verbleibt geschützt auf Endgerät und wird nicht geteilt
- Unterstützte SSH Key Formate:
  - ED25519 (bevorzugt)
  - ECDSA
  - RSA (mind. 2048-bit)
  - DSA (nicht mehr empfohlen)

**DEMO**

- SSH Schlüsselpaar generieren (ED25519)
  - Folgenden Befehl in Terminal ausführen:  
`ssh-keygen -t ed25519 -C "<comment>"`
  - Dateiname und Speicherort angeben
  - Passphrase für Schlüssel einrichten (optional)
- Öffentlichen Schlüssel auf GitLab hochladen
  - Inhalt der erstellten .pub Datei kopieren
  - Navigieren **Startseite** → **Avatar** → **Edit profile** → **SSH Keys** (Sidebar)
  - **Add new key** auswählen
  - Öffentlichen Schlüssel in die Textbox kopieren
  - Optional **Usage Type** und **Expiration Date** festlegen
  - **Add key** auswählen

**DEMO**

# Clonen von Projekten

User Settings / SSH Keys

Search settings

## SSH Keys

SSH keys allow you to establish a secure connection between your computer and GitLab. SSH fingerprints verify that the client is connecting to the correct host. Check the [current instance configuration](#).

Your SSH keys 3

### Add an SSH key

Add an SSH key for secure access to GitLab. [Learn more](#).

**Key**

```
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIGkcePbmlo3PYxYHsrLZNFvYa97ZavFWhmuSAfrt79EF gitlab
```

Begins with 'ssh-rsa', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', 'ecdsa-sha2-nistp521', 'ssh-ed25519', 'sk-ecdsa-sha2-nistp256@openssh.com', or 'sk-ssh-ed25519@openssh.com'.

**Title**

Key titles are publicly visible.

**Usage type**

**Expiration date**

Optional but recommended. If set, key becomes invalid on the specified date.

**Add key** **Cancel**

Title	Key	Usage type	Created	Last used	Expires	Actions

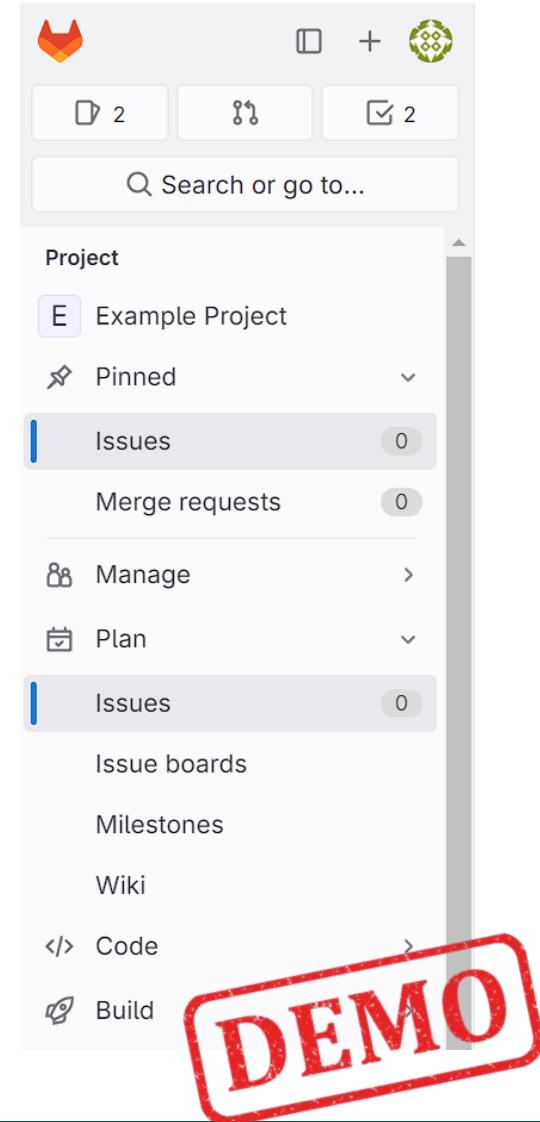


GitLab

# Planing Tools

# Issues

- In GitLab können innerhalb eines Projektes Issues für verschiedene Aufgaben angelegt werden
- Issues werden in der Menübar links sowohl als Shortcut als auch unter **Plan → Issues** angezeigt
- Issues können mit Labels versehen werden
  - Eigene Labels können beliebig unter **Manage → Labels** hinzugefügt werden
  - Default Set kann ebenfalls unter **Manage → Labels** generiert werden
  - Im Board können einzelne Listen für verschiedene Labels angelegt werden
  - Labels können priorisiert werden



# Labels

Example User / Example Project / Labels

All Subscribed

Search Name New label

Labels can be applied to issues and merge requests. Star a label to make it a priority label.

**Prioritized labels**  
Drag to reorder prioritized labels and change their relative priority.

No prioritized labels yet!  
Star labels to start sorting by priority.

**Other labels**

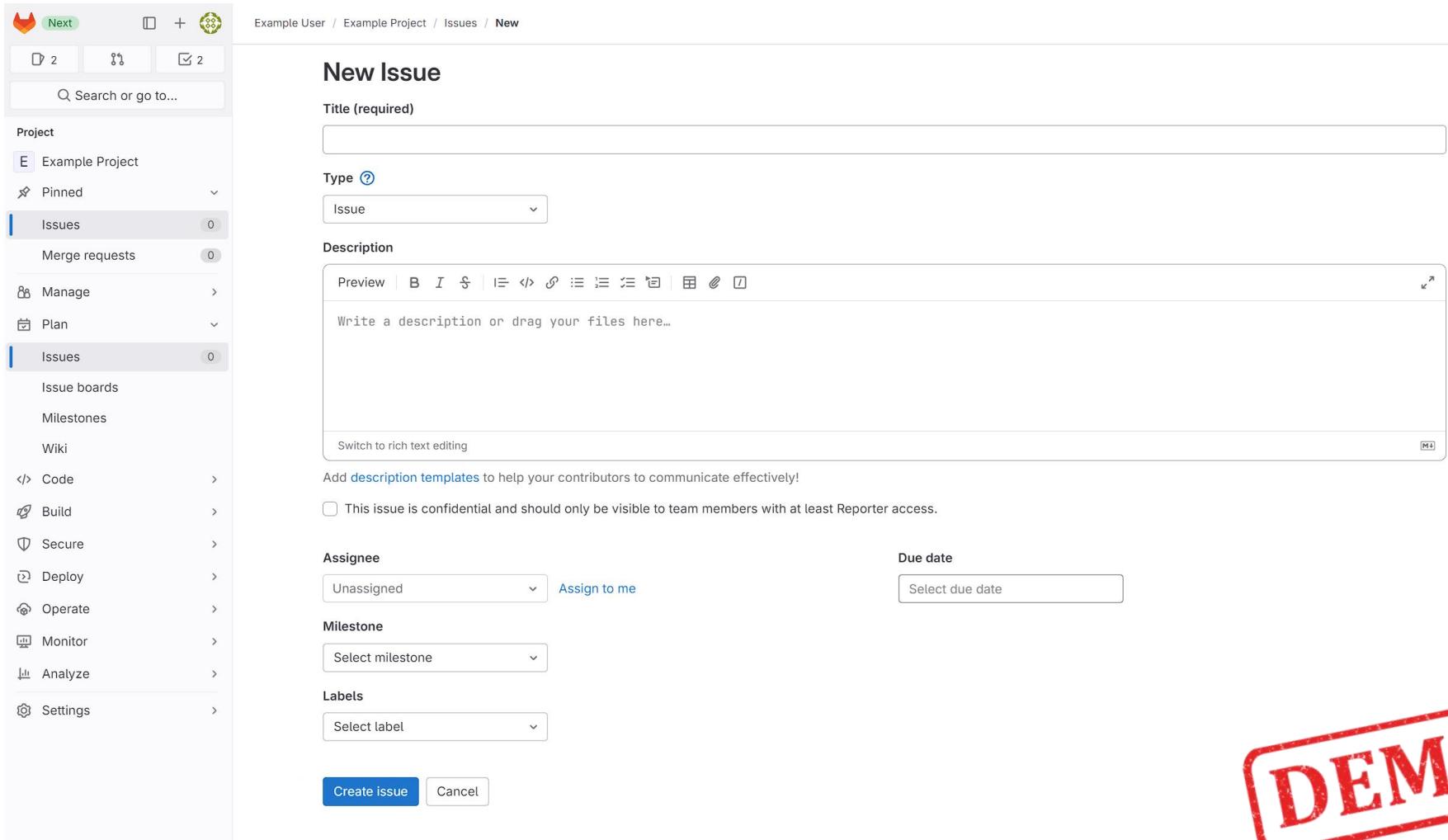
Label	Count	Issues	Merge requests	Subscribe	⋮
bug	0	Issues	Merge requests	Subscribe	⋮
confirmed	0	Issues	Merge requests	Subscribe	⋮
critical	0	Issues	Merge requests	Subscribe	⋮
discussion	0	Issues	Merge requests	Subscribe	⋮
documentation	0	Issues	Merge requests	Subscribe	⋮
enhancement	0	Issues	Merge requests	Subscribe	⋮
suggestion	0	Issues	Merge requests	Subscribe	⋮
support	0	Issues	Merge requests	Subscribe	⋮

Help

DEMO

# Issues

## Neues Issue anlegen



New Issue

Title (required)

Type ?

Issue

Description

Write a description or drag your files here...

Switch to rich text editing

Add [description templates](#) to help your contributors to communicate effectively!

This issue is confidential and should only be visible to team members with at least Reporter access.

Assignee

Unassigned [Assign to me](#)

Due date

Select due date

Milestone

Select milestone

Labels

Select label

Create issue Cancel

DEMO

# Issues

Example User / Example Project / Issues / #3

## Add unit tests

Open Issue created 7 minutes ago by Example User

Edit ⋮

Mark as done »

Assignee Edit  
Example User

Labels Edit  
testing x

Milestone Edit  
None

Weight  
This feature is locked. [Learn more](#)

Due date Edit  
None

Time tracking ⌚ +  
No estimate or time spent

Confidentiality Edit  
Not confidential

1 Participant

Move issue

**Activity**

- Example User added testing label 7 minutes ago
- Example User assigned to @Example\_User 7 minutes ago

Sort or filter

Preview | B I ♂ | ≡ ↔ ♂ ≡ ≡ ≡ ≡ | 田 📎  ⓘ  ⓘ | ✉

Write a comment or drag your files here...

Switch to rich text editing

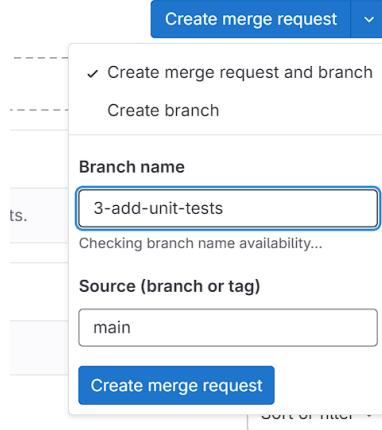
Make this an internal note  ⓘ

Comment ⋮ Close issue

DEMO

# Issues

- Issues können an Personen zugewiesen („assigned“) werden
- Milestones, Fälligkeitsdatum und Time Tracking können ebenfalls hinzugefügt werden
- GitLab bietet die Möglichkeit, direkt aus Issues Merge-Requests oder Branches anzulegen



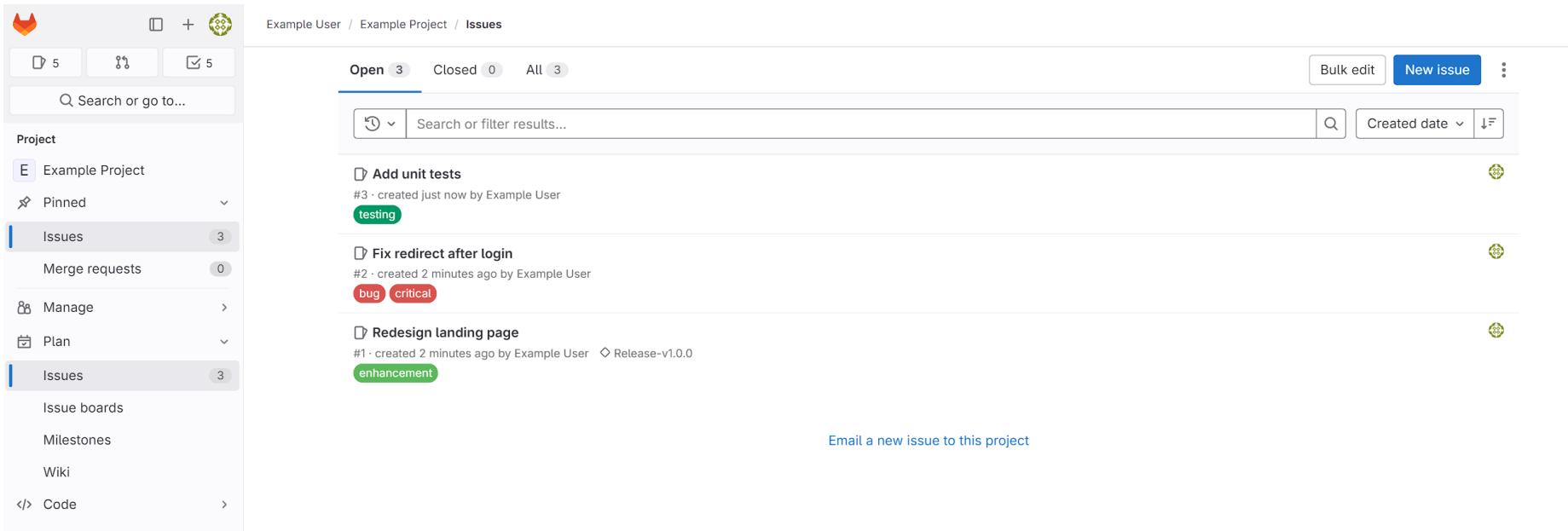
- Issues werden nach Erledigung geschlossen und damit in die „closed“ Sektion verschoben
- Templates für Issues können als Datei unter .gitlab/issue\_templates mittels Markdown in eine .md Datei gespeichert werden

**DEMO**

**DEMO**

# Issues

## Issue-Liste



The screenshot shows a GitLab interface for a project named "Example Project". The left sidebar includes links for "Project", "Merge requests", "Issues" (selected), "Issue boards", "Milestones", "Wiki", and "Code". The main area displays three open issues:

- Add unit tests** (#3) - Created just now by Example User **testing**. Status: Open.
- Fix redirect after login** (#2) - Created 2 minutes ago by Example User. Labels: **bug**, **critical**. Status: Open.
- Redesign landing page** (#1) - Created 2 minutes ago by Example User. Labels: **enhancement**. Status: Open.

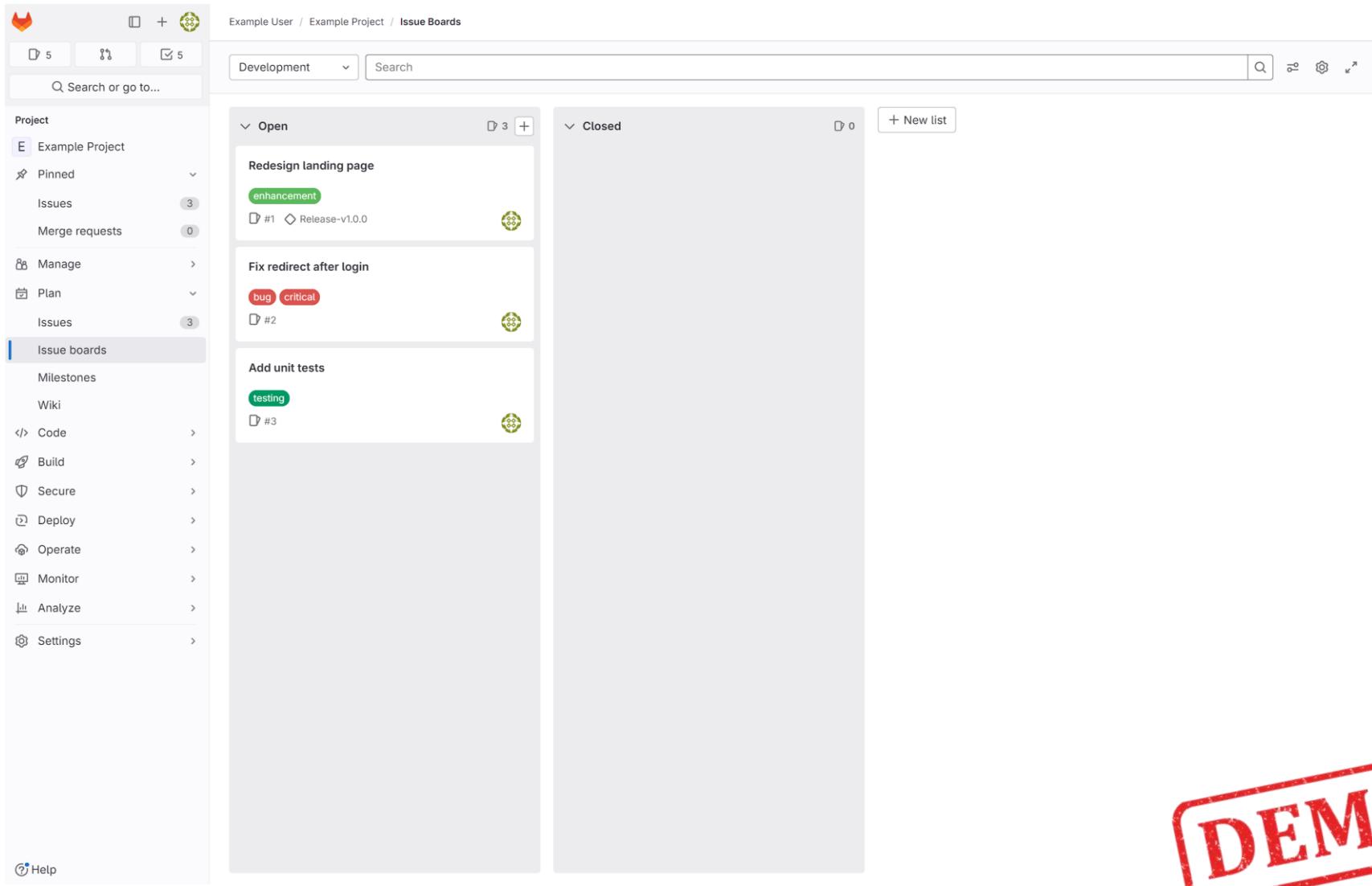
At the bottom right of the main area, there is a link to "Email a new issue to this project".

**DEMO**

- GitLab besitzt mit dem Issue Board ein Software Management Tool
- Kann zur Planung, Organisation und Visualisierung genutzt werden
  - Nutzung als Kanban oder Scrum Board
  - Effektive Aufgabenerfassung und Aufgabenverteilung
- Für Aufgaben werden einzelne Issues angelegt
  - Grundlegend in offene und geschlossen Labels aufteilt
- Board kann nach Labels strukturiert werden

**DEMO**

# Issue Board



The screenshot shows a GitLab interface for an 'Example Project'. The left sidebar is collapsed, showing sections like 'Project', 'Issues', 'Merge requests', 'Manage', 'Plan', 'Issue boards' (which is selected), 'Milestones', 'Wiki', 'Code', 'Build', 'Secure', 'Deploy', 'Operate', 'Monitor', 'Analyze', and 'Settings'. The main area displays an 'Issue Boards' view for the 'Development' project. At the top, there's a search bar with 'Development' selected and a 'Search' button. Below the search bar are two columns: 'Open' (containing 3 issues) and 'Closed' (containing 0 issues). A '+ New list' button is located at the top right of the closed column. The 'Open' column contains the following issues:

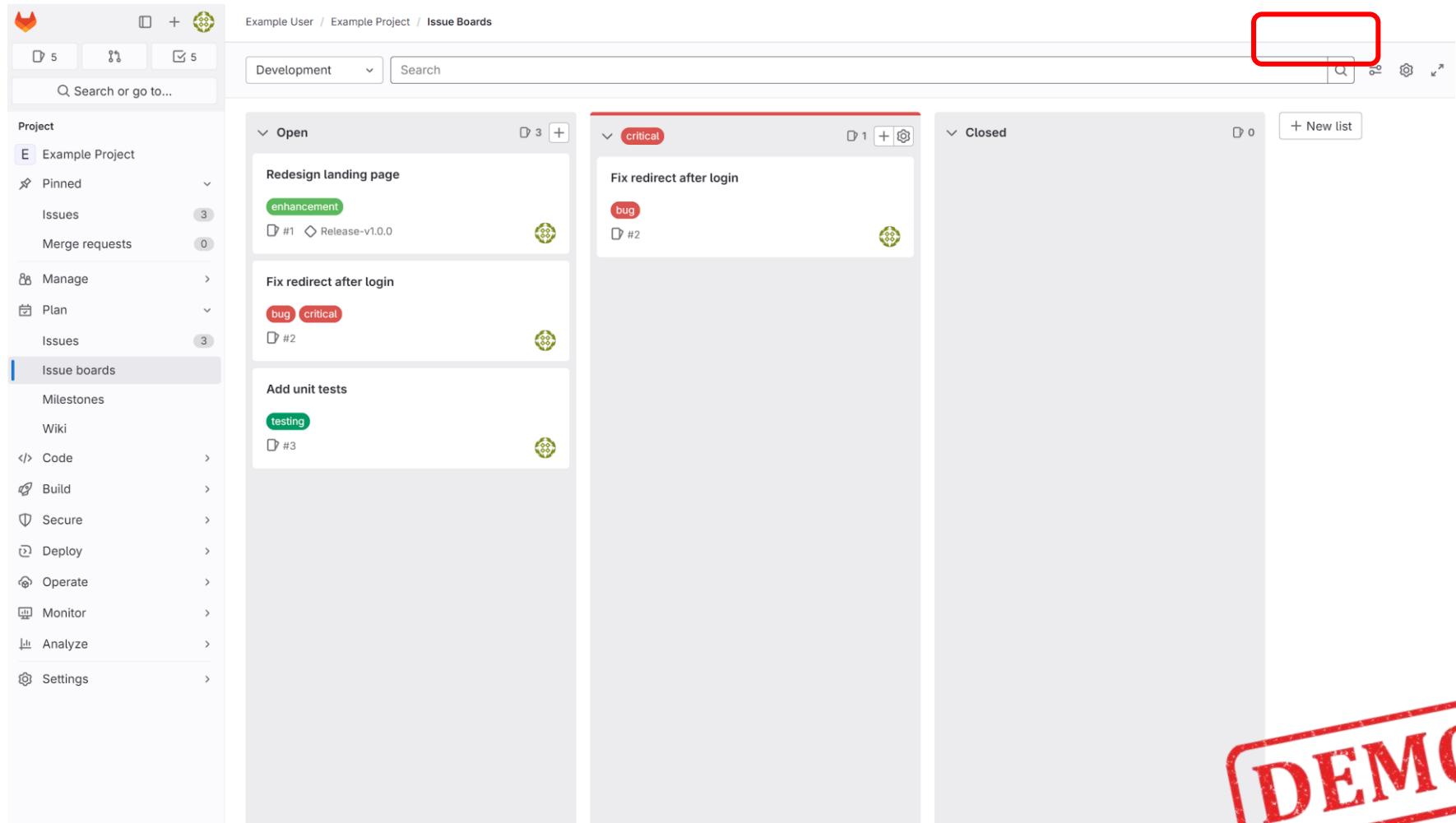
- Redesign landing page** (enhancement): #1, Release-v1.0.0
- Fix redirect after login** (bug, critical): #2
- Add unit tests** (testing): #3

At the bottom left of the main area, there's a 'Help' link.

**DEMO**

# Issue Board

## Neue Liste anlegen



The screenshot shows a GitLab Issue Board interface with the following structure:

- Open:** Contains three items:
  - Redesign landing page (enhancement)
  - Fix redirect after login (bug, critical)
  - Add unit tests (testing)
- Critical:** Contains one item:
  - Fix redirect after login (bug, critical)
- Closed:** Contains zero items.

A red rectangular box highlights the top right corner of the board area, likely indicating a 'New list' button or a placeholder for a new column.

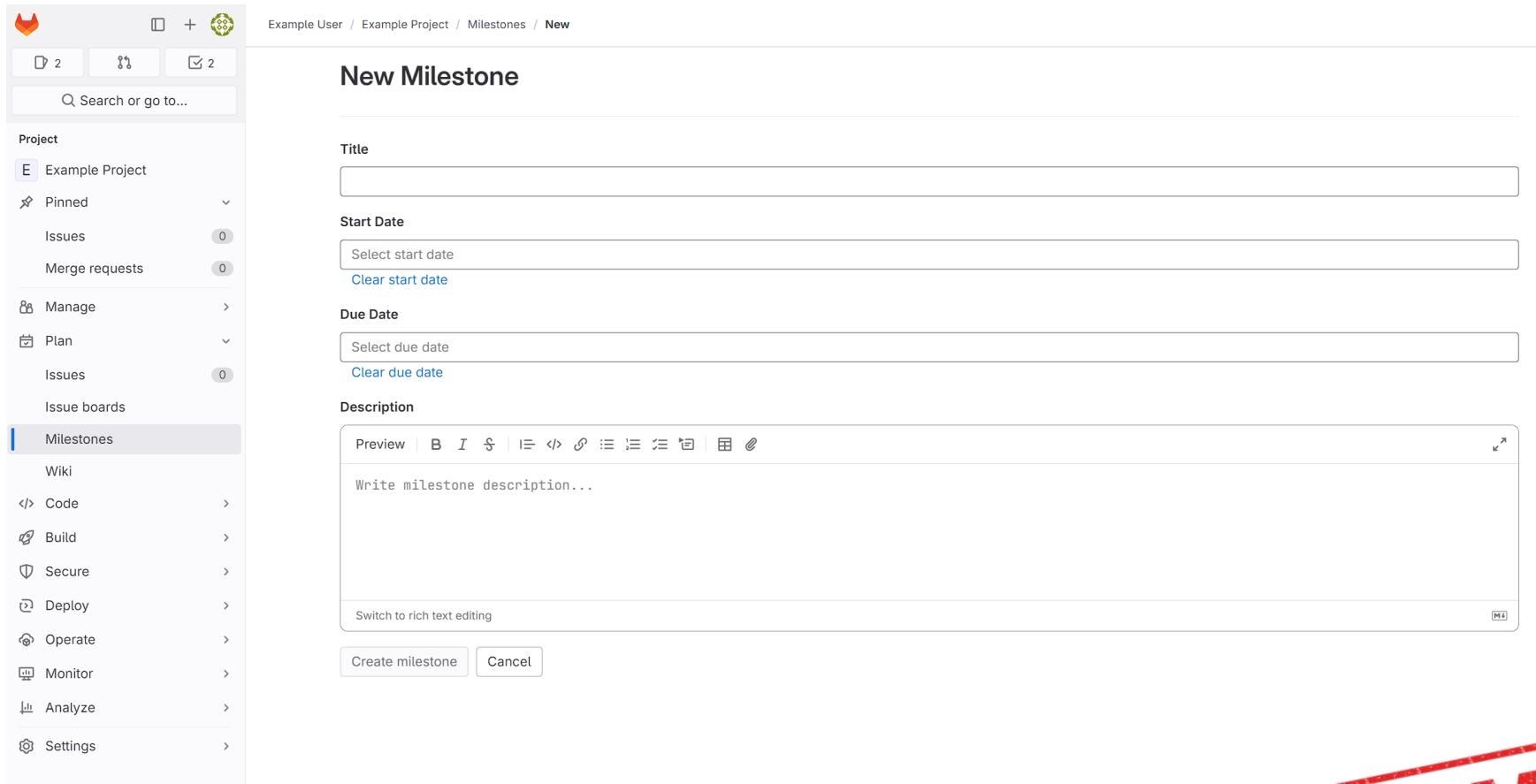
**DEMO**

# Milestones

- Milestones können als Ziel angelegt werden
- Issues und Merge-Requests können an Milestones gekoppelt und dadurch gruppiert werden
- Besitzen optionalen Start- und Endpunkt

**DEMO**

## Neuen Milestone anlegen



The screenshot shows the 'New Milestone' creation interface in GitLab. On the left, there's a sidebar with navigation links like 'Project', 'Issues', 'Merge requests', 'Manage', 'Plan', 'Issue boards', and 'Milestones' (which is currently selected). The main area has fields for 'Title', 'Start Date', 'Due Date', and a rich text editor for 'Description'. Buttons at the bottom are 'Create milestone' and 'Cancel'. A large red 'DEMO' stamp is overlaid in the bottom right corner.

Example User / Example Project / Milestones / New

### New Milestone

Title

Start Date

Due Date

Description

Preview | B I H | L <> S | M | C | R | E

Write milestone description...

Switch to rich text editing

Create milestone Cancel

- Ermöglicht Dokumentation innerhalb des Projekts
- Wird in einem separaten Git-Repository gespeichert
  - Keine direkte Versionierung zusammen mit dem Code
- Ist in einzelne Seiten aufgeteilt, welche in der Sidebar rechts angezeigt werden
- Ermöglicht Strukturierung in Unterseiten

DEMO

GitLab

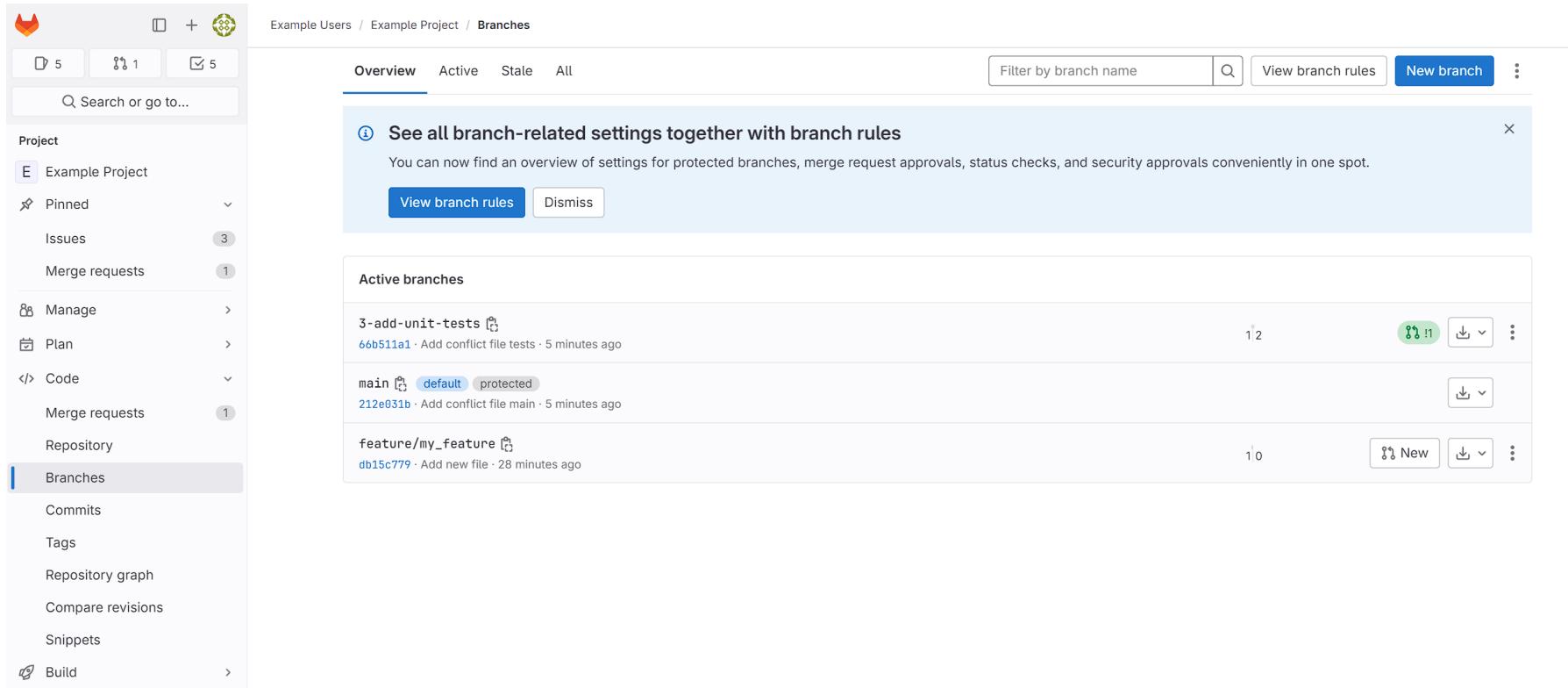
# Branches, Commits & Merging

DEMO

- Beim Erstellen eines Projektes legt GitLab eine Default Branch an
  - Einstellungen bezüglich Default Branch können auf Projekt, (Sub-)Gruppen oder Instanz Level vorgenommen werden
- Default Branch wird als *protected* bezeichnet und kann nicht gelöscht werden
  - Einstellungen hierzu können unter **Settings → Repository** feingranular vorgenommen werden
- Neue Branches können angelegt werden über
  - Projekt-Startseite
  - Unterpunkt Code → Branches
  - Über Issues
  - Lokal und per Push in GitLab übertragen
- Ist ein Branch abgeschlossen, wird üblicherweise ein Merge-Request (MR) gestellt, um Änderungen ins Projekt zu integrieren

**DEMO**

# Branches



The screenshot shows the GitLab interface for managing branches. On the left, there's a sidebar with project navigation: Project (Example Project), Issues (3), Merge requests (1), Manage, Plan, Code, Merge requests (1), Repository, and Branches (selected). Below these are Commits, Tags, Repository graph, Compare revisions, Snippets, and Build. At the top, there are filters for Overview, Active, Stale, All, a search bar, and buttons for View branch rules and New branch.

A modal window titled "See all branch-related settings together with branch rules" is open, explaining the new feature and providing "View branch rules" and "Dismiss" buttons.

The main content area displays "Active branches" with three entries:

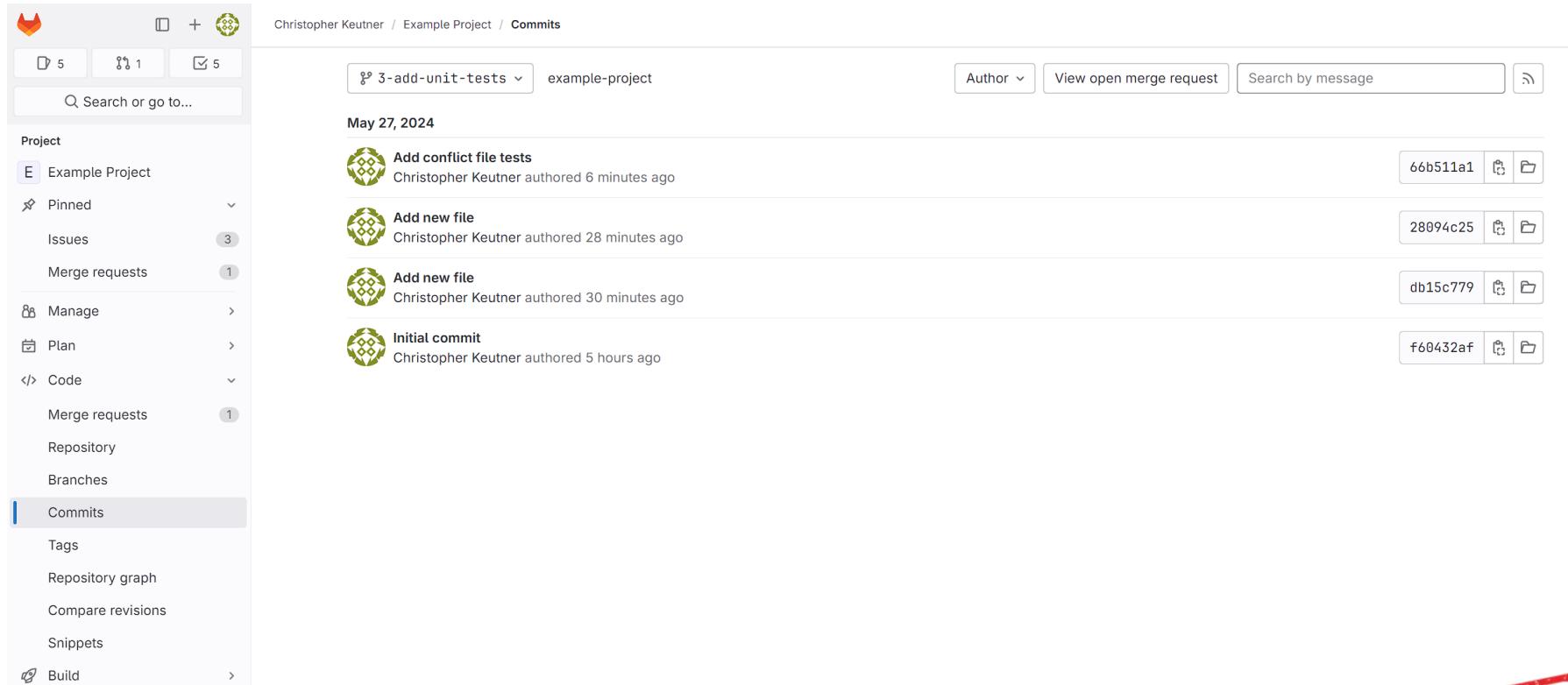
- 3-add-unit-tests · 66b511a1 · Add conflict file tests - 5 minutes ago
- main · 212e031b · Add conflict file main - 5 minutes ago
- feature/my\_feature · db15c779 · Add new file - 28 minutes ago

Each entry includes a green "New" button and a vertical ellipsis menu.

DEMO

# Commits

- Commits bietet die Möglichkeit, alle Commits einer ausgewählten Branches und deren Änderungen anzuzeigen



Christopher Keutner / Example Project / Commits

May 27, 2024

Commit Message	Author	Commit Hash
Add conflict file tests	Christopher Keutner authored 6 minutes ago	66b511a1
Add new file	Christopher Keutner authored 28 minutes ago	28094c25
Add new file	Christopher Keutner authored 30 minutes ago	db15c779
Initial commit	Christopher Keutner authored 5 hours ago	f60432af

DEMO

# Merge Requests

- Merge-Requests sind eine Anfrage, um neue Änderungen in das Projekt einzubringen
- Git-Merge von Branches als Grundkonzept
- Erweitert einfachen Merge um folgende Features
  - Beschreibung des Merge-Request
  - Anzeige der Codeänderungen für Reviews
  - Auflistung der einzelnen Commits des Requests
  - Verknüpfungen mit Issues oder Milestones
  - Kommentarsektion zur Diskussion
- Bietet Option den Source Branch nach dem Merge zu löschen
- Ermöglicht Squashing von Commits
  - Zielbranch enthält dann nicht alle einzelnen Commits des Source Branches und bleibt ggf. übersichtlicher

**DEMO**

# Merge Requests

Example User / Example Project / Merge requests / New

## New merge request

**Source branch**

Example\_User/example-project Select source branch

Select a branch to compare

**Target branch**

Example\_User/example-project main

Add new file Example User authored May 27, 2024 db15c779

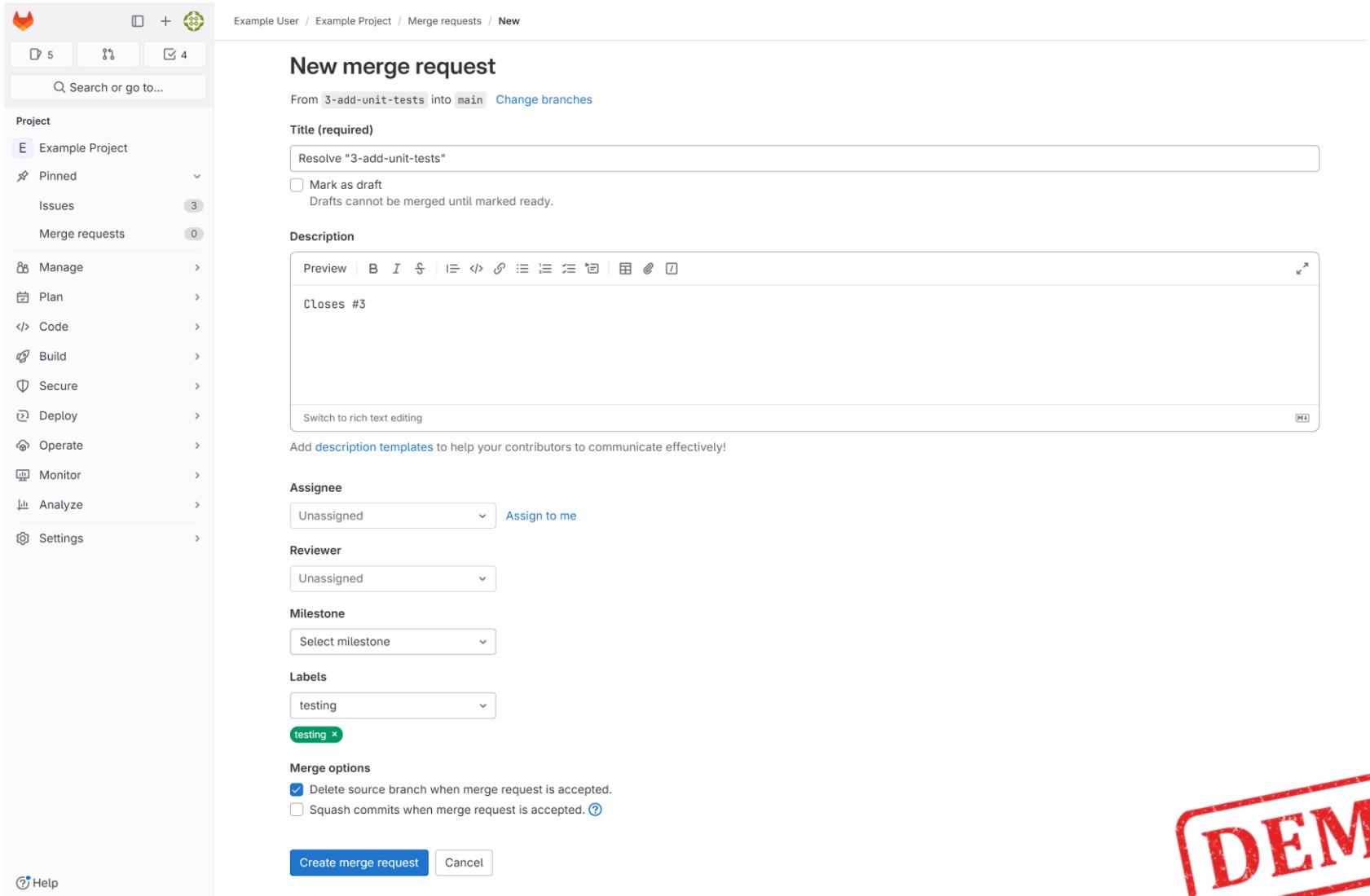
Compare branches and continue

Project

- E Example Project
- Pinned
- Issues 3
- Merge requests 0
- Manage >
- Plan >
- Code >
- Build >
- Secure >
- Deploy >
- Operate >
- Monitor >
- Analyze >
- Settings >

**DEMO**

# Merge Requests



The screenshot shows a 'New merge request' form in a GitLab interface. The left sidebar shows project navigation with sections like Project, Issues, Merge requests, and various operational tabs. The main form includes fields for Title (required), Description (with rich text editor preview), Assignee (Unassigned), Reviewer (Unassigned), Milestone (Select milestone), Labels (testing), and Merge options (Delete source branch when merge request is accepted, Squash commits when merge request is accepted). Buttons at the bottom are 'Create merge request' and 'Cancel'. A large red 'DEMO' stamp is overlaid in the bottom right corner.

Example User / Example Project / Merge requests / New

## New merge request

From 3-add-unit-tests into main [Change branches](#)

**Title (required)**

Resolve "3-add-unit-tests"

**Mark as draft**  
Drafts cannot be merged until marked ready.

**Description**

Closes #3

Switch to rich text editing

Add [description templates](#) to help your contributors to communicate effectively!

**Assignee**

Unassigned [Assign to me](#)

**Reviewer**

Unassigned

**Milestone**

Select milestone

**Labels**

testing

**Merge options**

Delete source branch when merge request is accepted.  
 Squash commits when merge request is accepted. [?](#)

[Create merge request](#) [Cancel](#)

- **Draft** – Merge Request ist für andere Personen einsehbar, kann aber noch nicht gemerged werden
- **Assignee** – Zugewiesene Person, die für den Merge-Request bzw. für das Einfügen der Änderungen zuständig ist
- **Reviewer** – Person(en), die die Änderungen reviewen und durch einen Approve bestätigen können. Approves können als verpflichtend sowie als Optional eingestellt werden
- **Milestone** – Zugehöriger Milestone
- **Labels** – Dienen zur Kategorisierung und Einordnung des MR
- **Merge Options**
  - Delete Source Branch → Source Branch wird nach Durchführung des MR gelöscht
  - Squash Commits → Commits des Source Branches werden gesquashed zum Target-Branch hinzugefügt

**DEMO**

# Merge Requests

- GitLab versucht durch Templates den MR bereits mit Informationen anzureichern
  - So führt die 3 beim Branch 3-add-unit-tests im vorherigen Beispiel dazu, dass in die Beschreibung ein Closes #3 eingefügt wird und die Labels des Issues übernommen werden
  - Generelle Templates können als Datei unter .gitlab/merge\_request\_templates mittels Markdown in eine .md Datei gespeichert werden
- Merge-Konflikte werden in der Übersicht angezeigt

- Merge blocked: 1 check failed

- Merge conflicts must be resolved.

[Resolve locally](#) [Resolve conflicts](#)

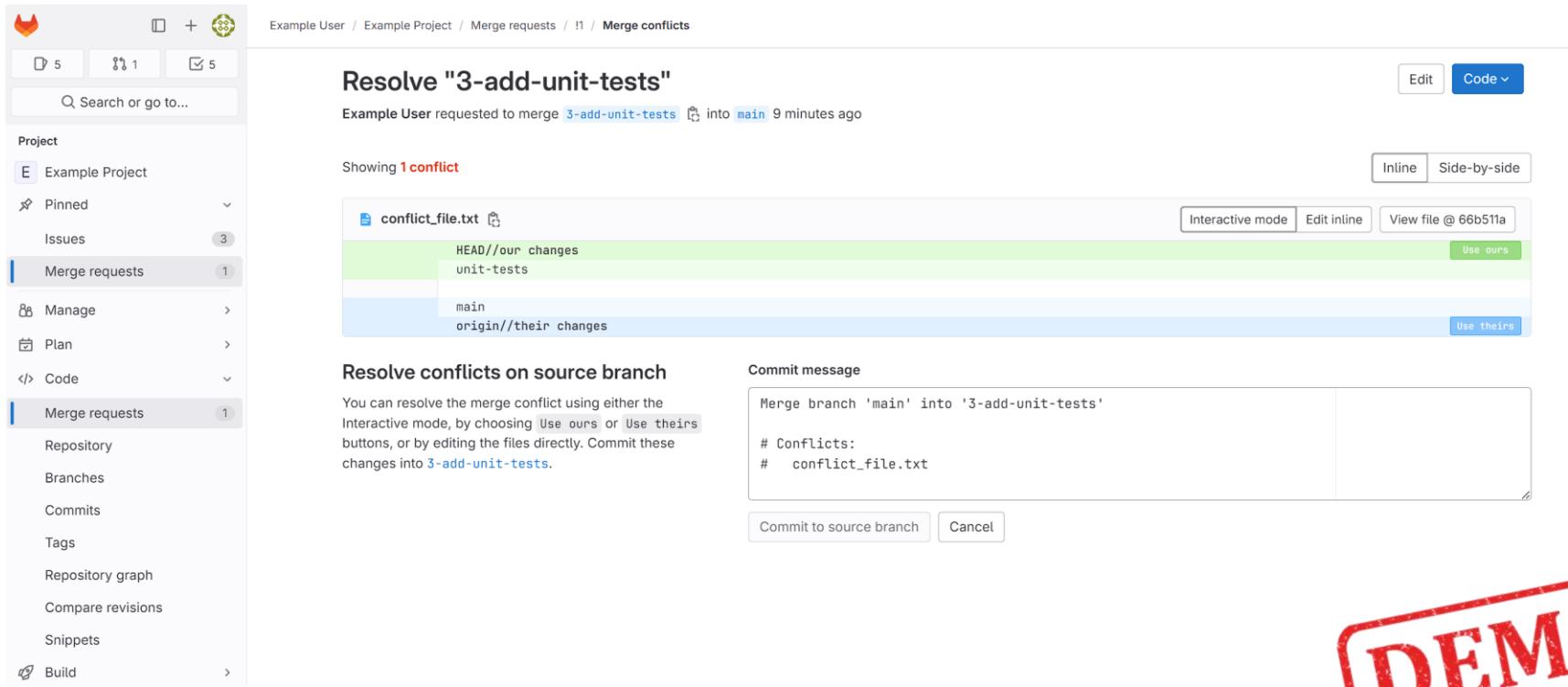
Merge details

- The source branch is [1 commit behind](#) the target branch.
- 1 commit and 1 merge commit will be added to `main` (squashes 2 commits).
- Source branch will be deleted.
- Closes issue [#3](#)

**DEMO**

# Merge Requests

- Auch bei Merge-Requests können, analog zu lokalen Konflikten, ebenfalls Merge-Konflikte auftreten
- Kleinere Konflikte können über GitLab aufgelöst werden, größere müssen jedoch lokal behoben werden



The screenshot shows the GitLab interface for resolving merge conflicts. On the left, the sidebar navigation includes 'Example Project' (selected), 'Merge requests' (1), and other options like 'Issues' (3), 'Commits', and 'Tags'. The main area displays a merge request titled 'Resolve "3-add-unit-tests"'. It shows a conflict in 'conflict\_file.txt' between 'HEAD//our changes' (green) and 'main' (blue). Buttons for 'Interactive mode', 'Edit inline', 'View file @ 66b511a', 'Use ours' (green), and 'Use theirs' (blue) are visible. Below this, sections for 'Resolve conflicts on source branch' and 'Commit message' are shown. The commit message box contains the text: 'Merge branch 'main' into '3-add-unit-tests'' and '# Conflicts: # conflict\_file.txt'. At the bottom are 'Commit to source branch' and 'Cancel' buttons.

**DEMO**

# GitLab Übung

## Aufgabe 1: Neues Projekt erstellen

1. (Optional) Beim Arbeiten mit GitLab ist es sinnvoll, mittels SSH Keys auf Remote Repositories zuzugreifen, statt immer wieder Username und Passwort zu verwenden.

Falls Sie noch keinen SSH Key in GitLab hinterlegt haben, so können Sie entweder die Anleitung aus GitLab nutzen oder dem Tutorial unter <https://docs.gitlab.com/ee/user/ssh.html> folgen.

2. Legen Sie ein neues privates Projekt **Playground** in Ihrem eigenen Bereich an, sodass nur Sie selbst Zugriff darauf haben. Achten Sie darauf, dass Sie das Repository mit einer Readme initialisieren, um dieses direkt klonen zu können.
3. Klonen Sie das Repository in ein lokales Verzeichnis auf Ihrem Rechner.

## Lösung Aufgabe 1: Neues Projekt erstellen

2.



### Create blank project

Create a blank project to store your files, plan your work, and collaborate on code, among other things.

Project name

Playground

Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, dashes, or spaces.

Project URL

<https://gitlab.com/>

Example\_User

Project slug

/ playground

Project deployment target (optional)

Select the deployment target

Visibility Level

Private

Project access must be granted explicitly to each user. If this project is part of a group, access is granted to members of the group.

Internal

The project can be accessed by any logged in user except external users.

Public

The project can be accessed without any authentication.

Project Configuration

Initialize repository with a README

Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.

Enable Static Application Security Testing (SAST)

Analyze your source code for known security vulnerabilities. [Learn more](#).

Create project



## Lösung Aufgabe 1: Neues Projekt erstellen

```
3. $ git clone git@gitlab.com:Example_User/playground.git
Cloning into 'playground'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
```

## Hinweis:

Man kann grundsätzlich viele Änderungen am Projekt sowohl *lokal* als auch über die *GitLab Weboberfläche* vornehmen. Für gewöhnlich bearbeitet man eher selten Dateien in der GitLab Weboberfläche, sondern nimmt Änderungen lokal in einer IDE vor und pusht diese danach ins Remote Repository.

Um das Arbeiten mit mehreren Entwicklern im Projekt etwas nachzustellen, werden wir im Folgenden auch Änderungen über die Weboberfläche vornehmen, um damit Änderungen von anderen Entwicklern zu simulieren.

Dadurch lassen sich verschiedene Szenarien erzeugen, deren Behandlung wir an unserem lokalen Repository üben werden.

## Aufgabe 2: Erste Schritte im Repository

1. Navigieren Sie in einem Terminal in das neue geklonte Projekt auf Ihrem Rechner.
2. Mit dem Befehl `git status` können Sie Ihren aktuellen Stand abfragen.  
Sie sollten sich auf dem Branch **main** befinden, der sich auf demselben Stand wie **origin/main** befindet.
3. `git status` vergleicht nur den lokalen Remote Tracking Bereich mit Ihrem aktuellen Branch. Es gibt keinen Aufschluss darüber, ob remote neue Änderungen verfügbar sind.  
Nutzen Sie `git fetch`, um ihren lokalen Remote Tracking Bereich mit dem GitLab Repository zu synchronisieren.  
Es sollten keine neuen Änderungen verfügbar sein.

## Lösung Aufgabe 2: Erste Schritte im Repository

1. \$ cd playground/

```
$ git status
On branch main
Your branch is up to date with 'origin/main'.
```

nothing to commit, working tree clean

2. \$ git fetch

## Aufgabe 3: Erste Datei hinzufügen

1. Legen Sie eine Datei **random\_numbers.sh** an und füllen Sie diese mit dem folgenden Inhalt

```
#!/bin/bash

random_number=$(( RANDOM % 10 + 1 ))

if [ $random_number -lt 11 ]; then
    echo "Number is less than 11"
fi
```

2. Committen Sie Ihre Änderungen auf dem **main** Branch.
3. Pushen Sie Ihre Änderungen ins Remote Repository. Vergewissern Sie sich über die Web-GUI, dass Ihr Push erfolgreich war.

## Lösung Aufgabe 3: Erste Datei hinzufügen

1. \$ nano random\_numbers.sh

2. \$ git add random\_numbers.sh

```
$ git commit -m "Add random_numbers.sh"
[main f7a7699] Add random_numbers.sh
 1 file changed, 7 insertions(+)
 create mode 100644 random_numbers.sh
```

3. \$ git push

Enumerating objects: 4, done.

Counting objects: 100% (4/4), done.

Delta compression using up to 16 threads

Compressing objects: 100% (3/3), done.

Writing objects: 100% (3/3), 415 bytes | 415.00 KiB/s, done.

Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

To gitlab.com:Example\_User/playground.git

bb2df46..f7a7699 main -> main

## Aufgabe 4: Issue erstellen

1. Über das Issue Board in GitLab können Aufgaben definiert und an bestimmte Personen zugewiesen werden.  
Legen Sie im Issue Board ein neues Issue **Add output if number greater than 5** an.  
Diese Aufgabe sollen Sie nun übernehmen, tragen Sie sich daher als Assignee ein.
2. GitLab bietet die Möglichkeit, aus Issues direkt Merge-Requests und Branches zu erstellen.  
Klicken Sie auf das Dropdown beim Button „Create merge request and branch“ und wählen Sie nur „Create branch“ aus.  
Erstellen Sie nun eine Branch, um die Änderungen für das Feature vorzunehmen.

## Lösung Aufgabe 4: Issue erstellen

### 1. New Issue

Title (required)

Add output if number greater than 5

Type 

Issue

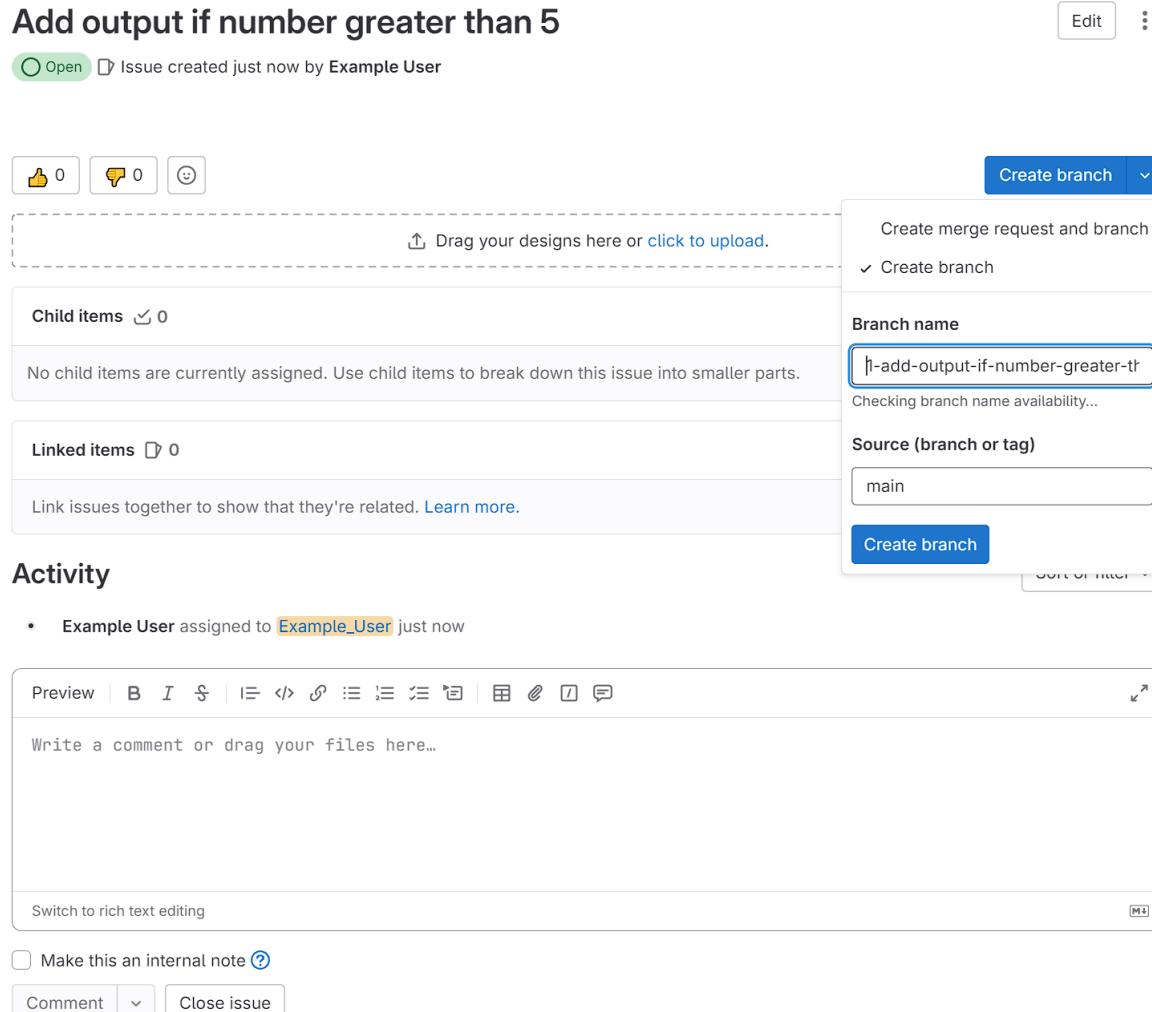
Description

Preview |           |                                  <img alt="comment icon

2.

## Add output if number greater than 5

 Open  Issue created just now by Example User



## Aufgabe 5: Lokale Änderungen

1. Führen Sie in Ihrem lokalen Repository den Befehl `git status` sowie `git branch -a` aus.  
Ihnen sollte der neu angelegte Branch aus GitLab nicht angezeigt werden, da Sie diesen erst vom Remote abrufen müssen.
2. Fetchen Sie alle Änderungen vom Remote.  
`git branch -a` sollte nun den neuen Branch anzeigen.
3. Wechseln Sie in den Feature Branch.
4. Öffnen Sie die `random_numbers.sh` Datei und nehmen Sie die geforderten Änderungen vor, indem Sie die Datei um ein weiteres *if-Statement* ergänzen, indem geprüft wird, ob die Zahl größer als 5 ist.
5. Committen Sie Ihre Änderungen und pushen Sie diese ins Repository.

## Lösung Aufgabe 5: Lokale Änderungen

1. \$ git status

On branch main

Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

\$ git branch -a

\* main

remotes/origin/HEAD -> origin/main

remotes/origin/main

2. \$ git fetch

From gitlab.com:Example\_User/playground

\* [new branch] 1-add-output-if-number-greater-than-5 -> origin/1-add-output-if-number-greater-than-5

\$ git branch -a

\* main

remotes/origin/1-add-output-if-number-greater-than-5

remotes/origin/HEAD -> origin/main

remotes/origin/main

## Lösung Aufgabe 5: Lokale Änderungen

```
3. $ git checkout 1-add-output-if-number-greater-than-5
Branch '1-add-output-if-number-greater-than-5' set up to track remote branch '1-
add-output-if-number-greater-than-5' from 'origin'.
Switched to a new branch '1-add-output-if-number-greater-than-5'
```

```
4. $ nano random_numbers.sh
```

```
$ cat random_numbers.sh
#!/bin/bash

random_number=$(( RANDOM % 10 + 1 ))

if [ $random_number -lt 11 ]; then
    echo "Number is less than 11"
fi
if [ $random_number -gt 11 ]; then
    echo "Number is greater than 5"
fi
```

## Lösung Aufgabe 5: Lokale Änderungen

```
5. $ git add random_numbers.sh

$ git commit -m "Add output if greater 5"
[1-add-output-if-number-greater-than-5 be535e7] Add output if greater 5
  1 file changed, 3 insertions(+)

$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 433 bytes | 433.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: To create a merge request for 1-add-output-if-number-greater-than-5,
visit:
remote:   https://gitlab.com/Example_User/playground/-
remote:   /merge_requests/new?merge_request%5Bsource_branch%5D=1-add-output-if-number-
remote:   greater-than-5
remote:
To gitlab.com:Example_User/playground.git
  f7a7699..be535e7  1-add-output-if-number-greater-than-5 -> 1-add-output-if-
number-greater-than-5
```

## Aufgabe 6: Merge Request

Nun wollen wir unsere Änderungen des Features in den main Branch übernehmen. Lokal würde man den Feature Branch in den main Branch mergen. Der Ablauf bei GitLab ist funktional sehr ähnlich, erweitert jedoch das reine Merging um einige Aspekte.

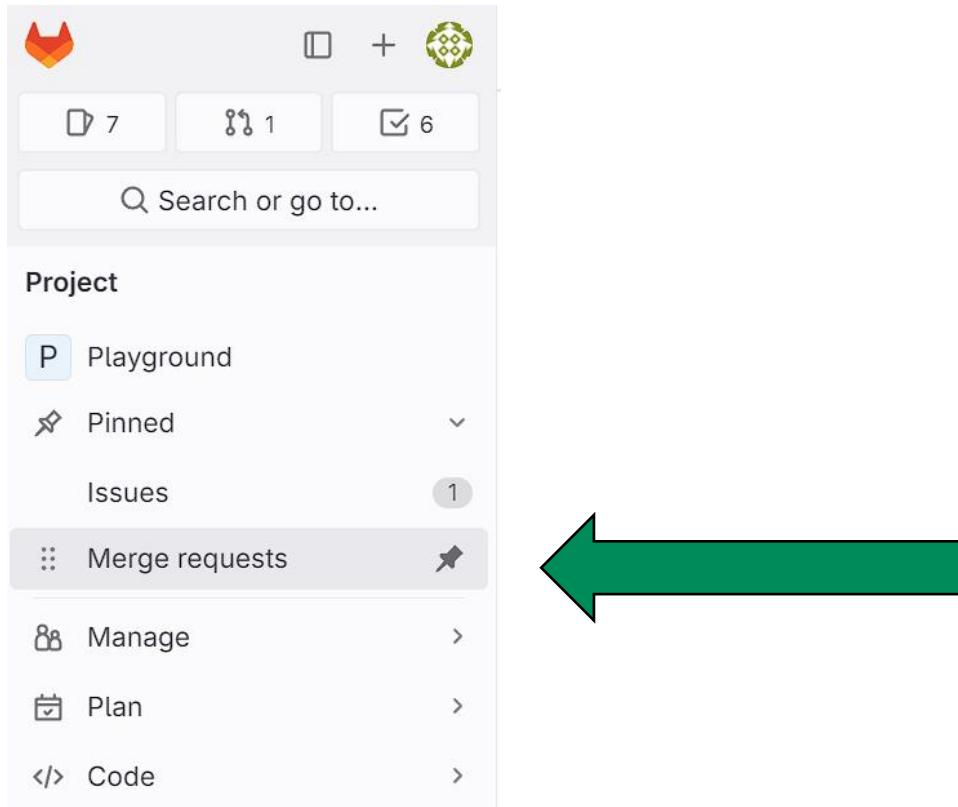
1. Navigieren Sie über die Sidebar in der GitLab Web-GUI auf den Punkt Merge Requests.
2. Erstellen Sie einen neuen Merge-Request, um Ihren Feature Branch in den **main** Branch zu mergen.  
Weisen Sie sich als Assignee und Reviewer zu (normalerweise würde man logischerweise jemand anderen als Reviewer eintragen).
3. Im Merge-Request haben Sie oben die Option, sich Commits und Changes anzuschauen. Reviewen Sie Ihre Änderungen und approven Sie den Merge-Request.

## Aufgabe 6: Merge Request

4. Mergen Sie nun den Merge Request mit der Option „Delete source branch“ und verifizieren Sie, dass die Änderungen auf dem **main** Branch vorhanden sind.
5. Löschen Sie Ihren lokalen Branch, da dieser abgeschlossen ist und keine Verwendung mehr besitzt.

## Lösung Aufgabe 6: Merge Request

1.



## Lösung Aufgabe 6: Merge Request

### 2. New merge request

#### Source branch

Example\_User/playground ▾

1-add-output-if-number-greate... ▾



Add output if greater 5

Example User authored Jun 12, 2024

be535e78



#### Target branch

Example\_User/playground ▾

main ▾



Add random\_numbers.sh

Example User authored Jun 12, 2024

f7a7699b



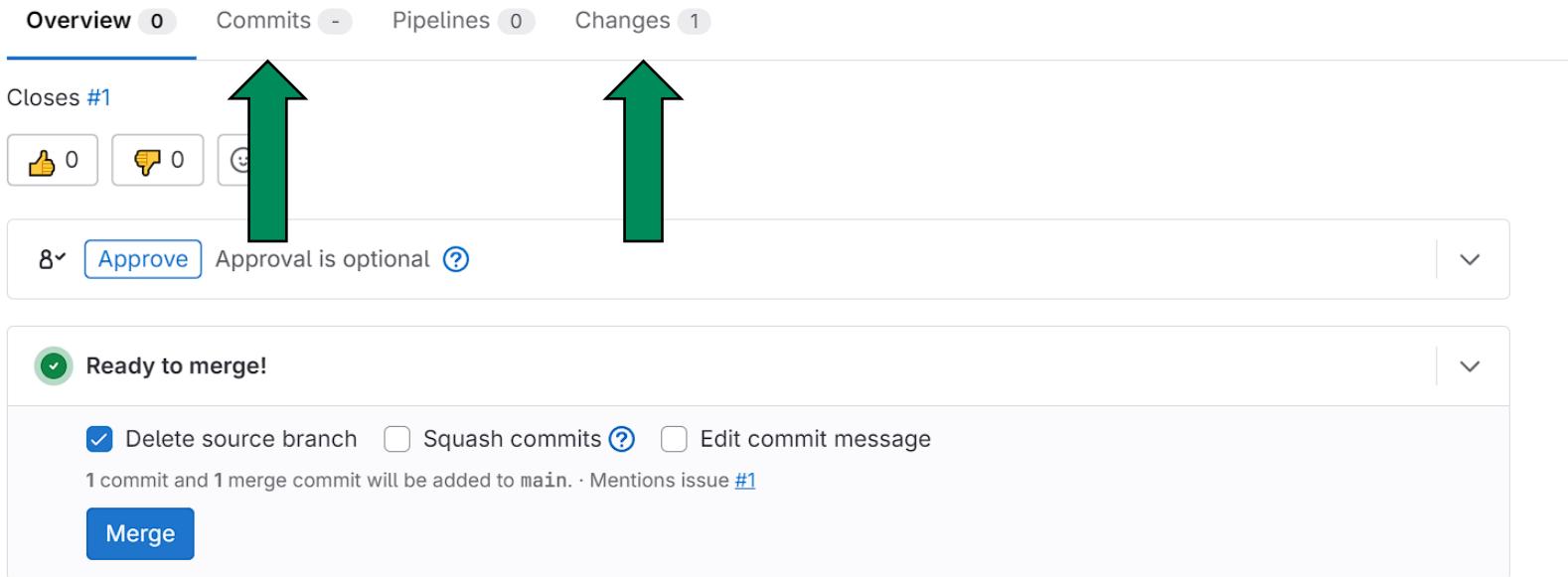
Compare branches and continue

# Lösung Aufgabe 6: Merge Request

## Lösung Aufgabe 6: Merge Request

### 3. Add output if greater 5

[Open](#) Example User requested to merge [1-add-output-if-number-gre...](#) into [main](#) just now



The screenshot shows a GitLab Merge Request interface. At the top, there are tabs for Overview (0), Commits (-), Pipelines (0), and Changes (1). Below the tabs, it says "Closes #1". There are three buttons: a thumbs up (0), a thumbs down (0), and a comment icon. A green arrow points upwards from the bottom of the page towards the thumbs up button. Below these are approval counts: 8 Approve and 0 Reject. A green arrow points upwards from the bottom of the page towards the "Approve" button. A note says "Approval is optional".

**Ready to merge!**

Delete source branch  Squash commits [?](#)  Edit commit message

1 commit and 1 merge commit will be added to main. · Mentions issue [#1](#)

**Merge**

## Lösung Aufgabe 6: Merge Request

### 4. Add output if greater 5

 Open Example User requested to merge [1-add-output-if-number-gre...](#)  into [main](#) just now

Overview 0 Commits 0 Pipelines 0 Changes 1

Closes #1



8v  Approve Approval is optional 

 Ready to merge!

Delete source branch  Squash commits   Edit commit message

1 commit and 1 merge commit will be added to main. · Mentions issue [#1](#)

Merge



## Lösung Aufgabe 6: Merge Request

```
5. $ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

$ git pull
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (1/1), 294 bytes | 294.00 KiB/s, done.
From gitlab.com:Example_User/playground
  f7a7699..fc1edde main      -> origin/main
Updating f7a7699..fc1edde
Fast-forward
  random_numbers.sh | 3 +++
  1 file changed, 3 insertions(+)

$ git branch -d 1-add-output-if-number-greater-than-5
Deleted branch 1-add-output-if-number-greater-than-5 (was be535e7).
```

## Aufgabe 7: Rebase von lokalen Branches

Nun wollen wir ein Rebase nutzen, um Remote Änderungen in unseren lokalen Branch zu integrieren.

1. Erstellen Sie ein neues Issue **Add output if number is maximum**. Dabei soll eine Ausgabe erfolgen, wenn die Random Nummer die größtmögliche Zahl ist.
2. Man muss nicht zwingend den Branch aus einem Issue heraus erstellen. Legen Sie dieses Mal lokal einen Branch **2-add-output-if-max** an und führen Sie die notwendigen Änderungen durch.
3. Committen Sie Ihre Änderungen und pushen Sie diese zum Remote Repository.

## Aufgabe 7: Rebase von lokalen Branches

In der Zwischenzeit hat ein Kollege die Aufgabe bekommen, den Zahlenbereich von 10 auf 1000 zu erhöhen.

Um die parallelen Änderungen von Ihm zu simulieren, führen wir diese über die Web-GUI durch. Ein zugehöriges Issue bzw. einen Branch überspringen wir zur Simplifizierung.

**Hinweis:** Ein gemeinsames Arbeiten über eine Datei sollte im Optimalfall vermieden werden, um Konflikte zu vermeiden.

4. Ändern Sie über GitLab den Random Number Bereich auf 1000 und committen Sie die Datei direkt aus dem Editor auf den **main** Branch.
5. Um im lokalen Repository die Änderungen zu übernehmen, müssen Sie den **main** Branch updaten. Wechseln Sie dazu in diesen und verwenden Sie `git pull`, um diese in Ihren lokalen **main** Branch zu übernehmen.

## Aufgabe 7: Rebase von lokalen Branches

6. Um die Änderungen nun auch in den Feature Branch zu übernehmen, müssen Sie in den Feature Branch wechseln und dort auf den **main** Branch rebasen.
7. Rebasen Sie Ihren Feature Branch auf den **main** Branch. Dabei sollten keine Konflikte auftreten, da sich die betreffenden Zeilen bei Ihnen nicht geändert haben.
8. Passen Sie Ihre Ausgabe bezüglich der Random Number an und committen Sie Ihre Änderungen.
9. Versuchen Sie Ihren Branch zum Remote zu pushen. GitLab wird hierbei den Push wegen nicht zueinander passenden Commit-Historien ablehnen. Pushen Sie daher Ihren Branch mit der Option **--force**.

## Aufgabe 7: Rebase von lokalen Branches

**Wichtig:** --force bewirkt, dass der Remote Branch durch die lokale Version überschrieben wird.

Falls andere Entwickler auf diesem Branch arbeiten würden, so könnten Sie weder push noch pull auf diesem ausführen, da die Commit-Historien nicht vereinbar sind. Ihre Branches wären damit kaputt und müssten komplett neu vom Remote abgerufen werden.

**Daher niemals auf Public Branches rebasen!**

## Lösung Aufgabe 7: Rebase von lokalen Branches

### 1. New Issue

Title (required)

Add output if number is maximum

Type [?](#)

Issue

Description

Preview | **B** *I* ~~S~~ | |

Write a description or drag your files here...

Switch to rich text editing

Add [description templates](#) to help your contributors to communicate effectively!

This issue is confidential and should only be visible to team members with at least Reporter access.

Assignee

Example User

Due date

Select due date

Milestone

Select milestone

Labels

Select label

Create issue

Cancel

## Lösung Aufgabe 7: Rebase von lokalen Branches

2. \$ nano random\_numbers.sh

```
$ cat random_numbers.sh
#!/bin/bash

random_number=$(( RANDOM % 10 + 1 ))

if [ $random_number -lt 11 ]; then
    echo "Number is less than 11"
fi
if [ $random_number -gt 11 ]; then
    echo "Number is greater than 5"
fi
if [ $random_number -eq 10 ]; then
    echo "Number is maximum value"
fi
```

## Lösung Aufgabe 7: Rebase von lokalen Branches

3. \$ git add random\_numbers.sh

```
$ git commit -m "Add max output"
[2-add-output-if-max ba27310] Add max output
 1 file changed, 4 insertions(+)

$ git push -u origin 2-add-output-if-max
Enumerating objects: 5, done.
...
To gitlab.com:Example_User/playground.git
 * [new branch]      2-add-output-if-max -> 2-add-output-if-max
Branch '2-add-output-if-max' set up to track remote branch '2-add-output-if-max'
from 'origin'.
```

## Lösung Aufgabe 7: Rebase von lokalen Branches

4.

P Playground 

main playground / +

History Find file Edit Code

Merge branch '1-add-output-if-number-greater-than-5' into 'main'  fc1ebedd Example User authored 11 minutes ago

Name	Last commit	Last update
README.md	Initial commit	41 minutes ago
random_numbers.sh	output if greater 5	24 minutes ago

Project information

- o 2 Commits
- 2 Branches
- 0 Tags
- 4 KiB Project Storage

README  
+ Add LICENSE  
+ Add CHANGELOG  
+ Add CONTRIBUTING  
+ Enable Auto DevOps  
+ Add Kubernetes cluster  
+ Set up CI/CD  
+ Add Wiki  
+ Configure Integrations

Created on  
June 12, 2024

random\_numbers.sh  README.md

Playground

Getting started

To make it easy for you to get started with GitLab, here's a list of recommended next steps.

Already a pro? Just edit this README.md and make it your own. Want to make it easy? [Use the template at the bottom!](#)

Add your files

## Lösung Aufgabe 7: Rebase von lokalen Branches



main playground / random\_numbers.sh

Find file Blame History Permalink

Update random\_numbers.sh  
Example User authored 7 minutes ago ab3cc3d8

random\_numbers.sh 265 B

Blame Edit Replace Delete

```
1 #!/bin/bash
2
3 random_number=$(( RANDOM % 1000 + 1 ))
4
5 if [ $random_number -lt 11 ]; then
6     echo "Number is less than 11"
7 fi
8 if [ $random_number -gt 11 ]; then
9     echo "Number is greater than 5"
10 fi
11
12
13
14
```

Open in Web IDE  
Edit single file  
Edit this file only.  
Open in Gitpod  
Launch a ready-to-code development environment for your project.  
Your workspaces  
A workspace is a virtual sandbox environment for your code in GitLab.  
To set up this feature, contact your administrator.



## Lösung Aufgabe 7: Rebase von lokalen Branches

Edit file

Write Preview changes

main random\_numbers.sh

No wrap

```
1 #!/bin/bash
2
3 random_number=$(( RANDOM % 1000 + 1 ))
4
5 if [ $random_number -lt 11 ]; then
6   echo "Number is less than 11"
7 fi
8 if [ $random_number -gt 11 ]; then
9   echo "Number is greater than 5"
10 fi
11
12
```



Commit message

Update random\_numbers.sh

Target Branch

main

Commit changes

Cancel

## Lösung Aufgabe 7: Rebase von lokalen Branches

```
5. $ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 320 bytes | 320.00 KiB/s, done.
From gitlab.com:Example_User/playground
  fc1edde..7ddba00  main      -> origin/main
Updating fc1edde..7ddba00
Fast-forward
  random_numbers.sh | 2 ++
  1 file changed, 1 insertion(+), 1 deletion(-)
```

## Lösung Aufgabe 7: Rebase von lokalen Branches

6. 

```
$ git checkout 2-add-output-if-max
Switched to branch '2-add-output-if-max'
Your branch is up to date with 'origin/2-add-output-if-max'.
```
  
7. 

```
$ git rebase main
Successfully rebased and updated refs/heads/2-add-output-if-max.
```

## Lösung Aufgabe 7: Rebase von lokalen Branches

8. \$ nano random\_numbers.sh

```
$ cat random_numbers.sh
#!/bin/bash

random_number=$(( RANDOM % 1000 + 1 ))

if [ $random_number -lt 11 ]; then
    echo "Number is less than 11"
fi
if [ $random_number -gt 11 ]; then
    echo "Number is greater than 5"
fi
if [ $random_number -eq 1000 ]; then
    echo "Number is maximum value"
fi

$ git add random_numbers.sh

$ git commit -m "Update max output"
[2-add-output-if-max 610db3e] Update max output
 1 file changed, 1 insertion(+), 1 deletion(-)
```

## Lösung Aufgabe 7: Rebase von lokalen Branches

```
9. $ git push --force
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 16 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 670 bytes | 670.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0), pack-reused 0
remote:
remote: To create a merge request for 2-add-output-if-max, visit:
remote:   https://gitlab.com/Example_User/playground/-
remote:   /merge_requests/new?merge_request%5Bsource_branch%5D=2-add-output-if-max
remote:
To gitlab.com:Example_User/playground.git
 + 4f455a1...610db3e 2-add-output-if-max -> 2-add-output-if-max (forced update)
```

## Aufgabe 8: Merge Konflikte

In dieser Aufgabe wollen wir uns das Auflösen von Merge-Konflikten anschauen.

Ein Kollege von Ihnen hat ebenfalls das Feature bezüglich Ausgabe bei maximaler Zahl umgesetzt und dieses schon in den **main** Branch integriert.

1. Simulieren Sie die Änderungen des Kollegen, indem Sie auch hier wieder über die Web-GUI direkt auf dem **main** Branch arbeiten.  
Kopieren Sie Ihre lokalen Änderungen bezüglich der Ausgabe und fügen sie diese über die Web-GUI in die **random\_numbers.sh** Datei ein.  
Verändern Sie dabei die Konsolenausgabe, sodass sich die Änderungen unterscheiden.
2. Committen Sie Ihre Änderungen direkt auf **main**.

## Aufgabe 8: Merge Konflikte

3. Stellen Sie für Ihren Feature Branch einen Merge Request. Nach der Erstellung sollte GitLab anzeigen, dass es Konflikte beim Mergen gibt.
4. Kleinere Konflikte lassen sich über die Web-GUI auflösen, größere Konflikte lassen sich nur lokal auflösen.  
Zur Übung lösen wir den Konflikt lokal auf.
5. Updaten Sie den **main** Branch auf den neusten Stand aus dem Remote Repository.
6. Lösen Sie die Konflikte nun, indem Sie entweder den **main** Branch in Ihren Feature Branch mergen oder indem Sie den Feature Branch auf den aktuellen Stand des **main** rebasen. Da Sie alleine auf Ihrem Branch arbeiten, bietet sich auch hier ein Rebbase an.
7. Lösen Sie den Konflikt, indem Sie sich für Ihre Ausgabe entscheiden.

## Aufgabe 8: Merge Konflikte

8. Pushen Sie nach Auflösen des Konflikts Ihre Änderungen ins GitLab.
9. Der Merge-Request sollte nun ohne Konflikte umsetzbar sein. Mergen Sie Ihr Feature in den **main** Branch, um die Änderungen abzuschließen.

## Lösung Aufgabe 8: Merge Konflikte

1.

Edit file

Write Preview changes

main | random\_numbers.sh No wrap

```
1 #!/bin/bash
2
3 random_number=$(( RANDOM % 1000 + 1 ))
4
5 if [ $random_number -lt 11 ]; then
6 | echo "Number is less than 11"
7 fi
8 if [ $random_number -gt 11 ]; then
9 | echo "Number is greater than 5"
10 fi
11 if [ $random_number -eq 1000 ]; then
12 | echo "Number is max"
13 fi
14
```

Commit message

Update random\_numbers.sh

Target Branch

main

2.

Commit changes Cancel

## Lösung Aufgabe 8: Merge Konflikte

### 3. 2 add output if max

 Open Example User requested to merge [2-add-output-if-max](#)  into [main](#) 10 minutes ago

Overview 0 Commits 1 Pipelines 0 Changes 1



0



0

8v [Approve](#)Approval is optional 

Merge blocked: 1 check failed

 Merge conflicts must be resolved.[Resolve locally](#)[Resolve conflicts](#)

## Merge details

- The source branch is [1 commit behind](#) the target branch.
- 1 commit and 1 merge commit will be added to [main](#).
- Source branch will be deleted.

## Lösung Aufgabe 8: Merge Konflikte

```
5. $ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

```
$ git pull
remote: Enumerating objects: 3, done.
...
From gitlab.com:Example_User/playground
  7ddba00..ab3cc3d  main      -> origin/main
Updating 7ddba00..ab3cc3d
Fast-forward
 random_numbers.sh | 3 +++
 1 file changed, 3 insertions(+)
```

## Lösung Aufgabe 8: Merge Konflikte

6. \$ git checkout 2-add-output-if-max  
Switched to branch '2-add-output-if-max'  
Your branch is up to date with 'origin/2-add-output-if-max'.

```
$ git rebase main
Auto-merging random_numbers.sh
CONFLICT (content): Merge conflict in random_numbers.sh
error: could not apply 241cba7... Update max output
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue".
hint: You can instead skip this commit: run "git rebase --skip".
hint: To abort and get back to the state before "git rebase", run "git rebase
abort".
Could not apply 241cba7... Update max output
```

## Lösung Aufgabe 8: Merge Konflikte

7. \$ nano random\_numbers.sh

```
$ cat random_numbers.sh
```

```
#!/bin/bash
```

```
random_number=$(( RANDOM % 1000 + 1 ))
```

```
if [ $random_number -lt 11 ]; then
    echo "Number is less than 11"
```

```
fi
```

```
if [ $random_number -gt 11 ]; then
    echo "Number is greater than 5"
```

```
fi
```

```
if [ $random_number -eq 1000 ]; then
    echo "Number is maximum value"
```

```
fi
```

```
$ git add random_numbers.sh
```

```
$ git rebase --continue
```

```
[detached HEAD 191992c] Update max output
```

```
 1 file changed, 1 insertion(+), 1 deletion(-)
```

```
Successfully rebased and updated refs/heads/2-add-output-if-max.
```

## Lösung Aufgabe 8: Merge Konflikte

```
8. $ git push --force
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 356 bytes | 356.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote:
remote: View merge request for 2-add-output-if-max:
remote: https://gitlab.com/Example_User/playground/-/merge_requests/2
remote:
To gitlab.com:Example_User/playground.git
 + 241cba7...191992c 2-add-output-if-max -> 2-add-output-if-max (forced update)
```

## Lösung Aufgabe 8: Merge Konflikte

### 9. 2 add output if max

 Open Example User requested to merge [2-add-output-if-max](#)  into [main](#) 19 minutes ago

Overview 0 Commits 1 Pipelines 0 Changes 1

 0

 0



8✓  Approval is optional 

 Ready to merge!

Delete source branch  Squash commits   Edit commit message

1 commit and 1 merge commit will be added to main.

