

# Anleitung zur Administration und Verwendung einer *gitcamp* - Schulinstanz

## Inhalt

1. Wichtige Begriffe und Rollen .....	1
2. Administration der <i>gitcamp</i> -Instanz .....	2
2.1. Nutzer_innen anlegen .....	2
2.1.1. Verarbeitung der Daten .....	2
2.1.2. CSV - Upload .....	3
2.2. Passwörter .....	4
2.2.1. Initialpasswort .....	4
2.2.2. Passwortreset .....	4
2.3. Weitere Infos .....	4
3. Organisationen und Teams .....	5
4. Repositories .....	8

## 1. Wichtige Begriffe und Rollen

Dieses Dokument soll als Schritt-für-Schritt-Anleitung zum Einrichten, Verwalten und Verwenden einer *gitcamp*-Instanz an Ihrer Schule dienen. Zum besseren Verständnis werden im Folgenden häufig verwendete Begriffe und Nutzerrollen spezifiziert.

- **Repository ("Repo"):** Der Ort, an dem Sie Ihren Code/ Ihre Dateien für ein Projekt einschließlich ihres Revisionsverlaufs speichern.
- **Organisation:** Gruppe von Nutzern mit Zugriff auf gemeinsame Repos → Klasse oder Kurs
- **Team:** Eine Organisation kann aus mehreren Teams mit unterschiedlichen Zugriffsrechten bestehen (normalerweise: "owners" und "students")
- **Schulverwalter\_in:** Zuständig für die Schulinstanz. Kann als einzige Person neue Nutzer (Lehrer\_innen oder Schüler\_innen) anlegen, löschen, Passwörter zurücksetzen etc.
- **Lehrerinnen und Lehrer:** Können neue Organisationen (Klassen) mit Teams und Organisations-Repos anlegen.
- **Schülerinnen und Schüler:** Können eigene Repos anlegen, aber keine Organisationen (Klassen) oder Teams

## 2. Administration der *gitcamp*-Instanz

### 2.1. Nutzer\_innen anlegen

Das Anlegen von Nutzerinnen und Nutzern geschieht grundsätzlich durch Hochladen von ASV-Daten im CSV - Format. Im folgenden sehen Sie eine Beispieltabelle für Lehrkräfte.

Gruppen	Vornamen	Nachname	ID	E-Mail
7a	Peter	Muster	43443fg-45354	<a href="mailto:peter@muster.de">peter@muster.de</a>
7a, IMP8	Ute	Fischer	43443fg-45354	<a href="mailto:ute@fischer.de">ute@fischer.de</a>
8b	Lena	Schmidt	23432ab-98765	<a href="mailto:lena@schmidt.de">lena@schmidt.de</a>
9c	Jonas	Meier	56474cd-34678	<a href="mailto:jonas@meier.de">jonas@meier.de</a>
10a	Sophie	Keller	76453ef-12345	<a href="mailto:sophie@keller.de">sophie@keller.de</a>
IMP8	Max	Bauer	98345gh-56789	<a href="mailto:max@bauer.de">max@bauer.de</a>
7b, 9c	Laura	Wagner	34567ij-23456	<a href="mailto:laura@wagner.de">laura@wagner.de</a>

Und hier die entsprechende CSV-Datei als plain text:

```
"Gruppen";"Vornamen";"Nachname";"ID";"E-Mail"
"7a";"Peter";"Muster";"43443fg-45354";"peter@muster.de"
"7a, IMP8";"Ute";"Fischer";"43443fg-45354";"ute@fischer.de"
"8b";"Lena";"Schmidt";"23432ab-98765";"lena@schmidt.de"
"9c";"Jonas";"Meier";"56474cd-34678";"jonas@meier.de"
"10a";"Sophie";"Keller";"76453ef-12345";"sophie@keller.de"
"IMP8";"Max";"Bauer";"98345gh-56789";"max@bauer.de"
"7b, 9c";"Laura";"Wagner";"34567ij-23456";"laura@wagner.de"
```

Die Schüler-CSV sieht im Grunde genau so aus. Lediglich das Attribut *Gruppen* wird zu *Klasse*. Auf der Seite des Upload-Dialogs (s.u.) sind entsprechende Vorlagen hinterlegt.

#### 2.1.1. Verarbeitung der Daten

Diese Tabelle würde nach dem Upload als CSV-Datei folgendermaßen verarbeitet werden:

- **Vor- und Nachname** wird importiert, aus *Vorname.Nachname* wird der Username gebildet
- Die **Emailadresse** wird entsprechend zugeordnet, falls vorhanden, ansonsten Vergabe einer Platzhalteradresse
- **Klassen** - Zuordnung wird importiert, es werden automatisch entsprechende Klassen (Organisationen) mit Lehrern und Schülern gebildet

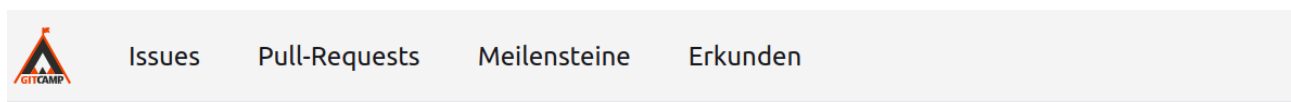
Dies aus der CSV übernommenen Vor- und Nachnamen werden:

- transponiert
- Umlaute werden entfernt
- doppelte Vornamen werden entfernt. Z.B.: aus "Ben Marlon MüllerHofholz" wird "Ben.MuellerHofholz"
- Der Benutzer jedes Nutzers Users wird nach folgendem Schema erzeugt: Vorname.Nachname
- Bei identischen Vornamen, Nachnamen und/oder Geburtsnamen jedoch unterschiedlicher ID wird an den generierten Nutzernamen eine fortlaufende Nummer angehängt.

Beim Import von Nutzern aus der CSV-Datei werden diese bereits in die richtigen Klassen (Organisationen) eingepflegt. Nutzer, die sowohl in der CSV, als auch in der DB vorhanden sind, werden bei Bedarf neuen Klassen (Neues Schuljahr) oder zusätzlichen Organisationen (zB neue Lerngruppe) zugeordnet. Vorhandene Nutzer (DB + CSV) werden aus allen, nicht in der CSV zu diesem jeweiligen Nutzer hinterlegten Klassen gelöscht.

### 2.1.2. CSV - Upload

Klicken Sie auf den Button *Benutzerverwaltung* (oben rechts). Sie gelangen zur Übersichtsseite der Schulverwalter-Funktionen.



## Schul-Verwalter Funktionen

### Benutzer einlesen

Hier können Sie schulische Benutzer hinzufügen. Sie können Schülerinnen und Schüler sowie Lehrkräfte anlegen. Für jede Klasse wird automatisch eine Organisation und ein Team in der Klassenorganisation angelegt.

[Zum CSV-Upload](#)

### Passwort ändern

Als Lehrkraft können Sie für Schülerinnen und Schüler das Passwort manuell zurücksetzen. Es wird Ihnen für die Schülerin oder den Schüler ein temporäres Passwort angezeigt, das bei der ersten Anmeldung geändert muss.

[Passwort ändern](#)

Über *Zum CSV-Upload* gelangen Sie nun zum Auswahldialog für den CSV - Upload:

# Upload der CSV-Datei

CSV-Datei für Lehrkräfte (GitCamp-Lehrkräfte)  Keine Datei ausgewählt.

CSV-Datei für Schülerinnen und Schüler (GitCamp-SuS)  Keine Datei ausgewählt.

CSV-Dateien hochladen

Abbrechen

Wählen Sie nun jeweils Ihre CSV-Datei mit den zu erstellenden Lehrer\_innen und Schüler\_innen aus. Bei erfolgreicher Verarbeitung Ihrer Datei, erhalten Sie nach wenigen Sekunden eine Übersicht über alle importierten Schüler\_innen und Lehrer\_innen.

**Achtung: Die CSV Dateien wird nur mit ";" als Trennzeichen erkannt und verarbeitet. Manche Tabellenkalkulationen speichern standardmäßig mit ",". Im Zweifelsfall entsprechende Einstellungen vornehmen bzw. die Datei im einfachen Texteditor öffnen und bearbeiten. Bitte im utf-8 Format abspeichern und alle Texte in der Datei in Hochkommas setzen.**

## 2.2. Passwörter

### 2.2.1. Initialpasswort

Beim Import wird automatisch ein Initialpasswort generiert, welches bei der ersten Anmeldung geändert werden muss. Die Lehrkraft erhält per E-Mail eine Liste mit den Nutzerdaten der Klasse(n) in denen sie unterrichtet. Außerdem wird die Passwortliste unmittelbar nach dem Upload angezeigt. Es wird empfohlen, an dieser Stelle direkt einen Screenshot zu erstellen, da der Versand per Mail Stand 10/2024 nicht immer zuverlässig funktioniert hat.

### 2.2.2. Passwortreset

Der Controller spielt je nach Lehrkraft die von ihr unterrichteten Klassen (Organisationen) in der GUI aus. Nach Auswahl der Klasse (Organisation) zeigt der Controller die Schülerinnen und Schüler der Klasse (Organisation) an. Nun kann die Lehrkraft in der jeweiligen Spalte der Schülerin oder des Schülers den Button „Passwort zurücksetzen“ anklicken. Anschließend wird der Lehrkraft ein temporäres Passwort angezeigt, welches bei der nächsten Anmeldung geändert werden muss.

## 2.3. Weitere Infos

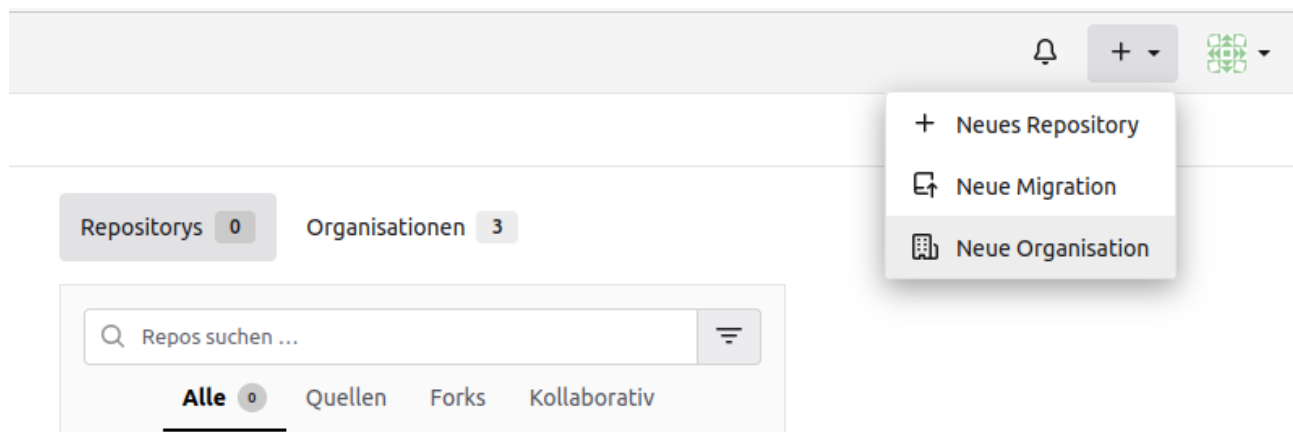
Weitere technische Informationen z.B. zu Löschroutinen und User-Quotas gibt es hier: [Kurzdokumentation IBBW](#).

### 3. Organisationen und Teams

Die *Organisation* ist ein zentrales Element der *gitcamp*-Umgebung. Sie stellt eine Gruppe von Usern mit Zugriff zu gemeinsamen Repositories dar. Im Schulumfeld werden Organisationen verwendet um z.B. Klassen, Kurse und AG's abzubilden. Eine Organisation besteht aus mindestens einem *Team*. Das sind Untergruppen von Usern mit genau spezifizierten Zugriffs- und Erstellungsrechten innerhalb der Organisation. Nur Lehrer\_innen dürfen Organisationen und Teams erstellen. Ob Schüler\_innen Organisations-Repositories erstellen dürfen, lässt sich in den Team-Einstellungen spezifizieren. Standardmäßig ist dies nicht der Fall.

Im Folgenden zeigt Ihnen die fiktive Nutzerin "Lilo Wagner" (Rolle: Lehrerin) schrittweise das Erstellen und Verwalten einer Organisation. Bitte beachten Sie, dass eine Organisation für Ihre Klasse / Ihren Kurs u.U. schon automatisch durch den Schulverwalter im Zuge des CSV-Uploads erstellt wurde (siehe oben).

Wählen Sie über den "+"-Button oben rechts "Neue Organisation".



Wählen Sie einen geeigneten Namen für Ihre Organisation.

### Neue Organisation

**Name der Organisation \***

Organisationsnamen sollten kurz und einprägsam sein.

**Sichtbarkeit \***

☐ Öffentlich

☒ Begrenzt (nur für authentifizierte Benutzer sichtbar)

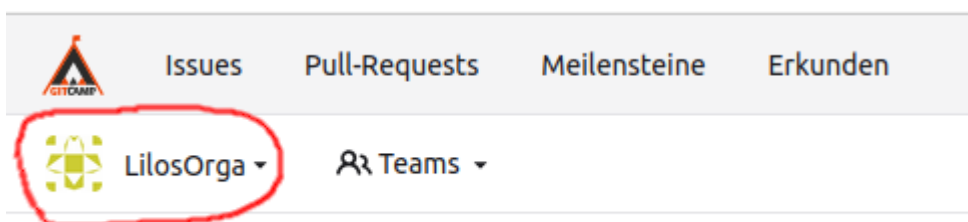
☐ Privat (nur für Organisationsmitglieder sichtbar)

**Berechtigungen**

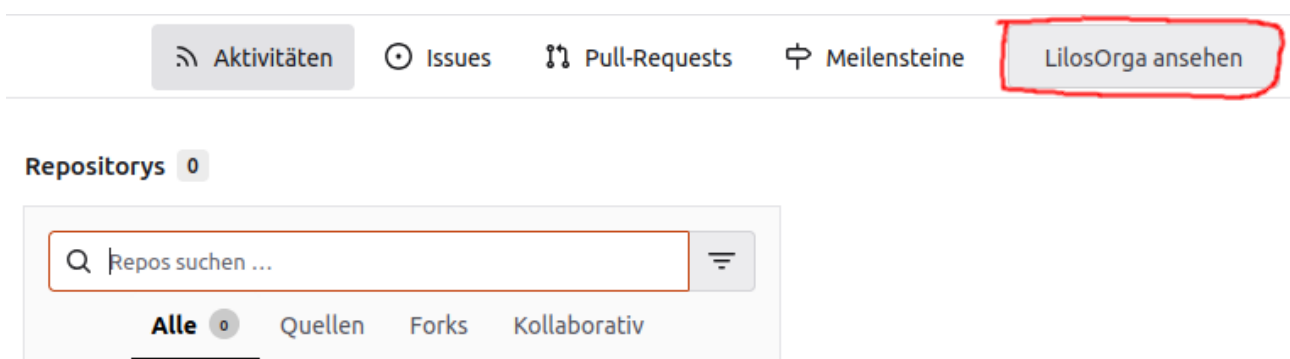
☒ Der Repository-Administrator kann den Zugriff für Teams hinzufügen und zurückziehen

[Organisation erstellen](#)

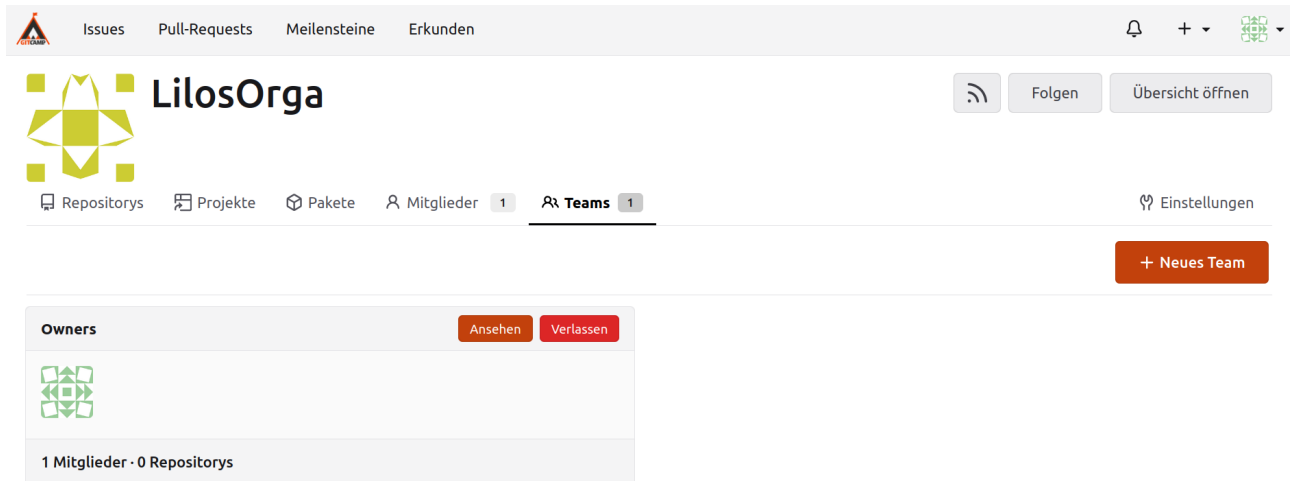
Die Sichtbarkeit Ihrer Organisation sollte in der Regel "begrenzt" (jeder eingeloggte User, also praktisch "schulweit") oder "privat" (nur für Orga-Mitglieder) sein. Nach der Erstellung sollte sich die Ansicht verändert haben. Wie links oben zu sehen, befinden wir uns nun in der Organisationsansicht. Über den kleinen Pfeil können Sie die Ansicht jederzeit zwischen allen Organisationen und Ihrer User-Ansicht wechseln.



Mit dem Button oben rechts gelangen Sie nun in die Detailansicht der Organisation:



Im folgenden Screenshot wurde der "Teams"-Tab ausgewählt. Erstellt man, eine Organisation, wird automatisch ein Team namens "Owners" erzeugt mit dem Erstellenden als einzigem Mitglied. Bei einer vom Schulverwalter per CSV Upload automatisch erstellten Organisation gibt es im Normalfall außerdem bereits ein Team namens "Students" mit den entsprechend Schüler\_innen.



Über den Button "Neues Team" kann man manuell ein weiteres Team erstellen. Dabei lassen sich umfangreiche Einstellungen bzgl. der Berechtigungen der Teammitglieder vornehmen. Diese sind gut beschrieben und sollten daher selbsterklärend sein. Ansonsten verweisen wir auf den entsprechenden Abschnitt der offiziellen Codeberg-Dokumentation: <https://docs.codeberg.org/collaborating/create-organization/#teams>.

Über *Ansehen* → *Einstellungen* ändern Sie diese Berechtigungen für ein bereits bestehendes Team. Dort können Sie das betreffende Team bei Bedarf auch löschen.

Zum Abschluss fügen wir dem neu erstellten Team noch zwei Mitglieder hinzu. Auf der Organisationsseite gelangen wir beim neu erstellten Team "Schueler" über *ansehen* auf folgender Seite:

**Schueler**
Beitreten

Dieses Team hat keine Beschreibung

**Zugriff auf das Repository**

- Bestimmte Repositorys

Einheit	Berechtigungen
Code	Lesen
Issues	Lesen
Pull-Requests	Lesen
Releases	Lesen
Wiki	Lesen
Zugriff auf externes Wiki	Lesen
Zugriff auf Externe Issues	Lesen
Projekte	Lesen
Pakete	Lesen

Einstellungen

0 Mitglieder
0 Repositorys

ts
Teammmitglied hinzufügen

ts-testuser1  
ts-testuser2  
ts-testuser3

Über das Textfeld werden nach Eingabe von mindestens zwei Zeichen entsprechende User vorgeschlagen, die man nun dem Team hinzufügen kann. Dabei findet man nur User, die unter *Einstellungen* → *Profil* die Benutzersichtbarkeit auf "öffentlich" geändert haben. Dazu klickt man zunächst auf den eigenen Avatar ganz oben rechts.

## 4. Repositories

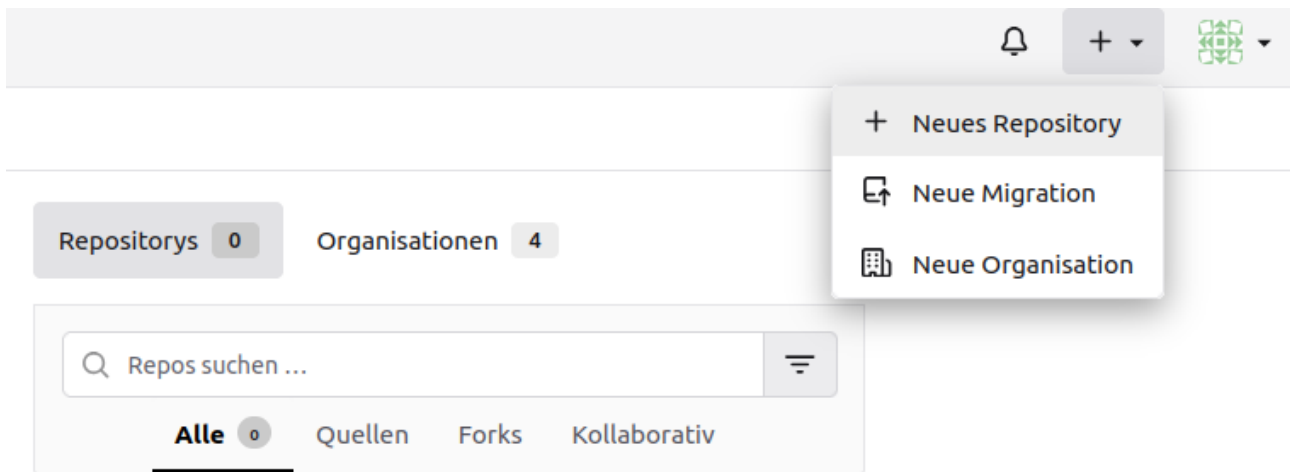
In den Repositories liegen Ihre Daten, also z.B. Ihr Quellcode oder Ihre Dokumentation. Dort geschieht auch die Versionsverwaltung für Ihr Projekt. Grundsätzlich gibt es zwei Arten von Repositories:

- Repos für persönliche Projekte
- Repos für Organisations-Projekte

Ein persönliches Projekt wäre z.B. ein Tool, das eine Informatiklehrkraft entwickeln möchte oder ein Programmierprojekt eines Kursteilnehmenden. Organisationsrepos stellen beispielweise einer Klasse Programmier-Vorlagen samt Dokumentation zur Verfügung oder beinhalten gemeinsame Projektarbeiten von Teams aus Schüler\_innen.

So erstellen Sie ein neues Repository:





Dabei öffnet sich diese Seite:

The image shows the 'Neues Repository' form. At the top, the title 'Neues Repository' is centered. Below it, a message reads: 'Ein Repository enthält alle Projektdateien inklusive der Revisionshistorie. Bereits woanders gehostet? Repository migrieren.' The form has four main sections: 1. 'Besitzer \*' with a dropdown menu showing 'Lilo.Wagner' and a note: 'Einige Organisationen könnten in der Dropdown-Liste nicht angezeigt werden, da die Anzahl an Repositorys begrenzt ist.' 2. 'Repository-Name \*' with a text input field containing 'LilosRepo' and a note: 'Ein guter Repository-Name besteht normalerweise aus kurzen, unvergesslichen und einzigartigen Schlagwörtern.' 3. 'Sichtbarkeit' with a checked checkbox 'In privates Repository umwandeln' and a note: 'Nur der Besitzer oder Organisationsmitglieder mit entsprechender Berechtigung werden in der Lage sein, es zu sehen.' 4. 'Beschreibung' with a text area containing 'Tolle Dateien' and a redacted area below it.

Es gibt noch weitere Einstellungen, die im Normalfall nicht geändert werden müssen. Falls doch, sei erneut auf die offizielle Dokumentation verwiesen: <https://forgejo.org/docs/latest/user/first-repository/>. *Besitzer* kann entweder der User selbst oder eine Organisation sein. So erstellt man entweder ein privates oder ein Organisationsrepo.

**Lilo.Wagner / LilosRepo**
Privat

<> Code
Issues
Aktivität

Beobachten beenden 1
 Favorisieren 0
 Einstellungen

Kurzanleitung

**Dieses Repository klonen**
 Benötigst du Hilfe beim Klonen? Öffne die [Hilfe](#).

Neue Datei
Datei hochladen
HTTPS
SSH

📄

**Erstelle ein neues Repository von der Kommandozeile aus**

```
touch README.md
git init
git checkout -b main
git add README.md
git commit -m "first commit"
git remote add origin https://fortbildner.gitcamp-bw.de/Lilo.Wagner/LilosRepo.git
git push -u origin main
```

**Bestehendes Repository via Kommandozeile pushen**

```
git remote add origin https://fortbildner.gitcamp-bw.de/Lilo.Wagner/LilosRepo.git
git push -u origin main
```

Die folgende Seite zeigt nun ein leeres Repository und enthält den Link auf das selbige zur weiterverwendung in der Konsole oder einem sonstigen git-fähigen Tool. Außerdem steht ein Anleitung zur Verfügung, mit der man entweder ein neues oder ein bestehendes lokales Repository verknüpfen kann.

Man kann auch im *gitcamp* selbst neue Dateien erstellen und zu einem Repository hinzufügen. Auch wenn diese Funktion voraussichtlich eher selten zum Einsatz kommen wird, zeigen wir sie zum Abschluss kurz, um einen Blick auf die Versionshistorie einer Datei werfen zu können.

Wir klicken auf *Neue Datei* und füllen dann folgendermaßen aus:

**Lilo.Wagner / LilosRepo**
Privat

<> Code
Issues
Pull-Requests
Aktivität







LilosRepo / 
📘 oder Abbrechen

<> Neue Datei
👁 Vorschau

```
1 Dies ist ein Text!
```

Bestätigen wir nun weiter unten mit *Änderungen committen*, haben wir die erste Datei unseres Repos erstellt. Klicken Sie nun auf das Stift-Symbol, um die Datei zu ändern und fügen sie einen

weiteren Satz hinzu. Nach erneutem *Commit* sollte das Ganze in etwa so aussehen:

2 Commits				
Autor	SHA1	Nachricht	Datum	
 Lilo.Wagner	369d590b32	datei1.md aktualisiert	vor 2 Minuten	 
 Lilo.Wagner	95425ae4cc	datei1.md hinzugefügt	vor 3 Minuten	 


Klickt man nun auf die zweite, geänderte Version (oben), erhält man detaillierte Informationen, über die Änderungen im Vergleich zur Vorversion:

**datei1.md aktualisiert**

Quellcode durchsuchen

Operationen ▾


...

 **Lilo.Wagner** vor 5 Minuten

Ursprung **95425ae4cc** Commit **369d590b32**

± 1 geänderte Dateien mit 1 neuen und 1 gelöschten Zeilen

>≡ +↕- ...

▼ 2  datei1.md 

Unescapen Datei anzeigen

@ -1 +1 @@

1 - Dies ist ein Text!

1 + Dies ist ein Text! Und was für einer...