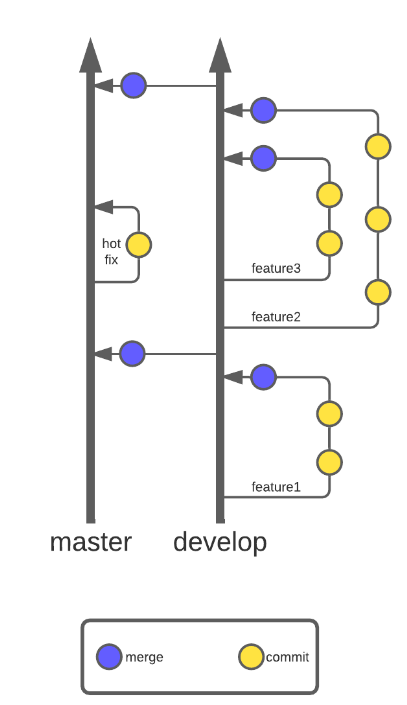
Collaboration auf Github

Die Zusammenarbeit der Entwickler erfolgt über das Code-Repository GitHub.

# Branches

Der main-Branch enthält den Projektcode im aktuellen Deploy-fertigem Zustand. Dieser Branch läuft auf dem Server und wird daher immer nur upgedatet, wenn ein neues Release fertig ist. Dann wird der develop-Branch in den main-Branch gemerged.

Die Entwickler arbeiten nicht direkt in diesem Branch und können auch nicht in diesen Branch pushen. Die Entwicklung schreitet im develop-Branch fort. Jedoch kann auch nicht direkt in diesen Branch gepusht werden. Für jedes Feature wird vom jeweiligen Entwickler ein neuer Branch, für genau dieses eine Feature, vom develop-Branch abgezweigt. Nach Fertigstellung des Features wird auf GitHub Pull Request, von dem Feature Branch in den develop-Branch gestellt. Dieser muss von mindestens einem anderen Collaborator genehmigt werden (Code Review Approval), und muss alle Tests bestanden haben. 

# Features

Features sollten als möglichst kleine Einheiten definiert werden. Idealerweise sollte es immer möglich sein, einen Feature Branch am Morgen eines Tages zu eröffnen, das Feature an diesem Tag zu entwickeln und am Abend des Tages das fertige Feature in den develop-Branch zu mergen. Dies bringt den Vorteil, dass alle Entwickler\*innen immer auf einem sehr aktuellen Stand sind und dass es zu sehr wenigen Merge-Konflikten kommt. Sollten Feature Branches regelmäßig länger als zwei Tage offen sein, müssen Features in Zukunft in kleinere Häppchen eingeteilt werden.

# 3 Workflow

Folgend wird der Workflow beschrieben, der für eine erfolgreiche Zusammenarbeit unabdingbar ist und daher von allen Entwickler\*innen eingehalten werden soll.

Der Workflow orientiert sich an Scrum und wird in dieser Form in großen Unternehmen für agile Softwareprojekte eingesetzt, daher lohnt es sich für alle Studierende den Workflow einzuhalten, um Erfahrungen im Team zu sammeln.

Der gesamte Workflow findet mittels GitHub statt.

## 3.1 Projekt Boards

Für jedes Team existiert ein Projekt auf GitHub. Dies dient lediglich der Übersichtlichkeit, alle Entwickler\*innen haben Zugriff auf jedes Projekt und können und sollen teamübergreifend agieren, kommentieren und kollaborieren. Für jedes Projekt existiert ein Board, welches alle geplanten, aktiven und abgeschlossenen Tasks zeigt.

## 10.3.2 Issues

Jedes zu implementierende Feature wird in seine Tasks zerlegt, und diese werden auf GitHub als Issue erstellt. Jedes Issue sollte eine aussagekräftige Beschreibung haben und es sollte ein passendes Label gewählt werden.

Es ist auch sinnvoll innerhalb der Beschreibung eine Definition of done (DoD) zu erstellen, welche angibt, wann das Issue als erledigt gilt, um genau festzulegen, was exakt im Scope eines jeden Issues liegt.

Außerdem sollte jedes Issue dem passenden Projekt Board (Backend, Frontend etc.) zugeordnet werden, damit diese nicht untergehen. Sobald mit einem Task begonnen wird, muss das Issue im Board in die Spalte In Progress verschoben werden, und ein Assignee eingefügt werden, so kann jeder erkenn das ein Issue gestartet wurde und wer dafür zuständig ist.

## 3.3 Branches

Für jedes Issue wird, sobald es begonnen wird, ein eigener Branch eröffnet mit folgendem Namenschema:

*type/#issue-kurze-beschreibung*

, wobei issue die Ticketnummer des Issues ist und type eines von feature (Implementierung eines neuen Features), fix (Beheben eines Bugs/Fehlers), refactor (Umschreiben des Codes ohne neue Funktionalität) oder task (Alle anderen Arbeiten) ist.

Beispiel:

*feature/127-Adding-api-delete-endpoint*

Wäre der passende Branchname für die Implementierung eines Features, dessen Issue die Ticketnummer 127 hat und dessen Inhalt das Hinzufügen eines API-Endpoints zum Löschen von Elementen ist.

3.4 Commits

Die Commit-Messages in einen Branch müssen ebenfalls die Issue-Nummer enthalten und haben folgendes Schema:

*type: #issuenr Kurze Beschreibung*

Und optional eine längere Erklärung nach einer Leerzeile. Wichtig ist das Symbol #, da dann der Commit automatisch durch GitHub zum Issue zugeteilt wird.

## 3.5 Pull Requests

Sobald ein Issue gelöst wurde, wird ein Pull Request zum Mergen des Branches in den develop Branch erstellt. Der PR muss dann mindestens ein Approval bekommen und die Tests bestehen, um dann gemergt zu werden.

## 3.6 Probleme und Hilfe

Wir nutzen das GitHub Discussion Forum für Ankündigungen, Fragen, Probleme, Ideen und mehr.