Git Workflow

We are using git as a versioning system using the Git Branching Workflows

Main branches

- **main** - the stable version used in productive environments. It is the latest release version which is tested and, to our knowledge, bug-free.
- **develop** - here the latest development branches are joined when they are finished. At a release the develop branch is merged into the main branch.

The branches in which the work (in progress) is happening have following prefixes:

- **feature_** - e.g. feature_user_profile_images
- **fix_** - e.g. fix_map_bug

Workflow

If working on a new feature/fix:

- Start with a new branch from **develop** and call it *feature_something* or *fix_something*

```
git checkout develop
git checkout -b feature_something
```

- Implement changes, commit

```
git add .
git commit -m "Implemented feature something"
```

- Merge **develop** regularly to your branch to keep merge conflicts to a minimum but at least before merging back

```
git merge develop
```

- Make sure that all tests pass, ideally have 100% coverage and Mypy checks are passing too
- Document database changes (see below) and add translations if needed
- Merge your branch to **develop**

```
git checkout develop
git merge feature_something
```

Around once a month a new version is released where the **develop** branch is merged into the **main** branch.

Database changes

If there are database changes make sure that

- there is a comment with the issue number at the SQL update script (e.g. at install/upgrade/3.12.0.sql)
- that the issue is noted at util/changelog.py

That way it's easier to deal with updates in the **develop** branch when dealing with multiple new features which need database updates.