Mobile Computing Project

Inhalt

1 Project description

The Mobile Computing module consists of a project that you carry out as part of a team.

Project objective

The objective of the project is to design, develop, and implement a framework for operating and displaying a 5G network (RAN, core, and transport network). The user interface can be implemented as a GUI, web interface, or text-based interface. The format can be freely selected.

Functional scope

The framework should enable a complete network view. This includes, among other things:

a. Topology overview
   - Display of all network components: 5G core network functions, RAN, transport network, SDN controller (if applicable), etc.
   - Visible relationships between components (e.g., data paths, logical connections)
   - Overview of all relevant IP addresses, interfaces, and function modules

b. Control & configuration
   - Starting/stopping individual network functions/services or the entire network
   - Loading, adjusting, and saving configuration profiles (MNC, MCC, slices, bit rates, etc.)
   - Adjusting rules in the transport network (SDN/iptables/tc)
   - Activating prioritizations or bandwidth limitations
   - Network slicing

c. Network status & monitoring (Optional!)
   Information overview, e.g.:
   - Active PDU sessions
   - Data paths and routing information
   - Bit rates, latencies, packet losses
   - Active configurations of network functions
   - Status of sessions, interfaces, and services

Use case selection and network configuration

A key feature of the framework is the selection and combination of different use cases. Use cases describe typical 5G application scenarios, e.g.:

- Mobile voice and data services (classic smartphone use)
- Connected cars
- Industrial automation (industrial IoT)
- Emergency communication/mission-critical services
- Smart city applications
- eHealth & telemedicine

The framework is designed to enable multiple use cases to be activated simultaneously (e.g., IoT + Connected Cars). Based on the selection, the 5G network should automatically:

- be configured correctly,
- activate all necessary network functions,
- adjust priorities, bit rates, and data paths accordingly.

Automated verification and test cases

After selecting the use cases, the framework should trigger an automated network verification, e.g., by executing defined test cases (e.g., latency tests, throughput tests, session establishment).

Expected project results

At the end of the project, the following results should be available:

1. Software/prototype
   - Functional UI
   - Visualization of the network architecture
   - Configuration interfaces
   - Test automation module
   - Transport network control
2. Documentation
   - Architecture and design description
   - Implementation details
   - Description of all use cases and test cases
   - User manual for the UI
3. Presentation/demonstration
   - Demo of the UI
   - Sample configurations
   - Presentation of test automation results

2 Project procedure

Create and plan your project based on the guidelines provided here.

I. Project start: Team organization - Early collection of individual strengths and skills (e.g., technology, organization, design, research) in order to assign tasks appropriately. Establishment of communication rules - e.g., weekly team meetings, minute-taking, selection of suitable additional tools for GitHub (e.g., MS Teams).

II. Project planning (create project plan in GitHub according to instructions):

- definition of the project goal
  - formulate a clearly defined main goal.
  - define concrete results (deliverables) that must be available at the end.
- define work packages (each work package should contain):
  - goal description
  - expected result
  - effort estimate (time in hours/days)
  - dependencies (which tasks must be completed beforehand)
  - responsible person + substitute

Example work packages: research & requirements analysis, concept development, implementation/production, testing/quality assurance, documentation, presentation

- set milestones - milestones mark important progress points, e.g.:
  - project definition and role allocation completed
  - concept finalized
  - prototype/interim status
  - final version
  - delivery & presentation

III. Project implementation (at least weekly)

- weekly status meetings – e.g., covering the following points:
  - progress of work packages (verifiable through commits, not just source code)
  - obstacles/problems (e.g., can be mapped using issues in GitHub)
  - adjustment of the schedule (can be mapped in the roadmap, e.g., by completing issues)
  - open questions (issues can be labeled, for example)

Each meeting ends with, for example: an updated task list, binding to-dos for the week, minutes with responsibilities

IV. Project completion phase – finalization:

- completion of all work packages
- quality assurance based on predefined criteria
- review of documentation
- preparation of the presentation with demonstration (demonstration max. 20 minutes, min. 15 minutes)
- reflection within the team - recommended guiding questions:
  - What went well?
  - What was challenging?
  - Which distribution of roles was effective?
  - What should be done differently in the next project?

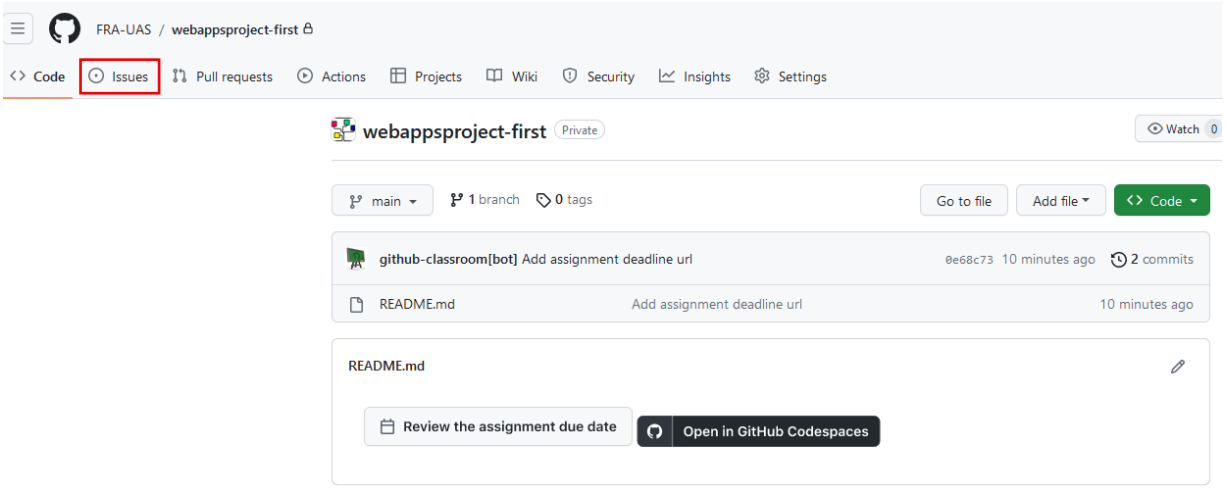
The following is required as documentation for the project and must be stored in GitHub:

- Design/architecture of the application system with all interfaces and APIs.
- Description and overview of the application system. Take advantage of the possibility to create diagrams for overviews and possibly also for process flows (e.g., sequence diagrams).
- GitHub repository must have a good structure organized into folders and files.

3 Project planning

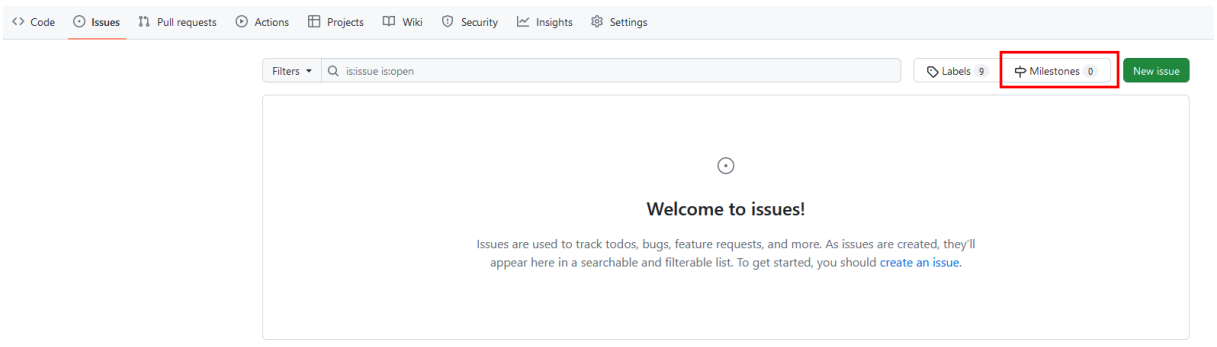
A central point of your project work in the team is project planning. This **always takes place at the start of the project** and is monitored and updated throughout the entire project duration. Planning is the basis for a successful project and, therefore, also for a corresponding evaluation. The following is a brief explanation of how to carry out the project planning.

The following image shows an overview of an exemplary repository.



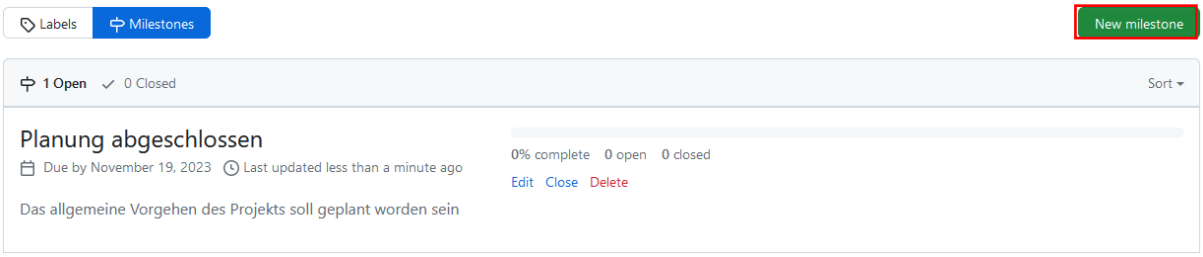
You will find the Issues tab (meaning in the the project is work packages) in the top menu bar. Here, you define all topics or problems that must be solved during the project. Create a work plan to track problems and discuss issues. With GitHub Issues, you can express ideas with GitHub Flavored Markdown, assign and mention contributors, respond with emojis, clarify with attachments and videos, and reference code such as commits, pull requests and deploys. With task lists, you can break down large issues into tasks, further organize your work with milestones and labels, and track relationships and dependencies.

Open the Issues view. And then switch to the milestones.

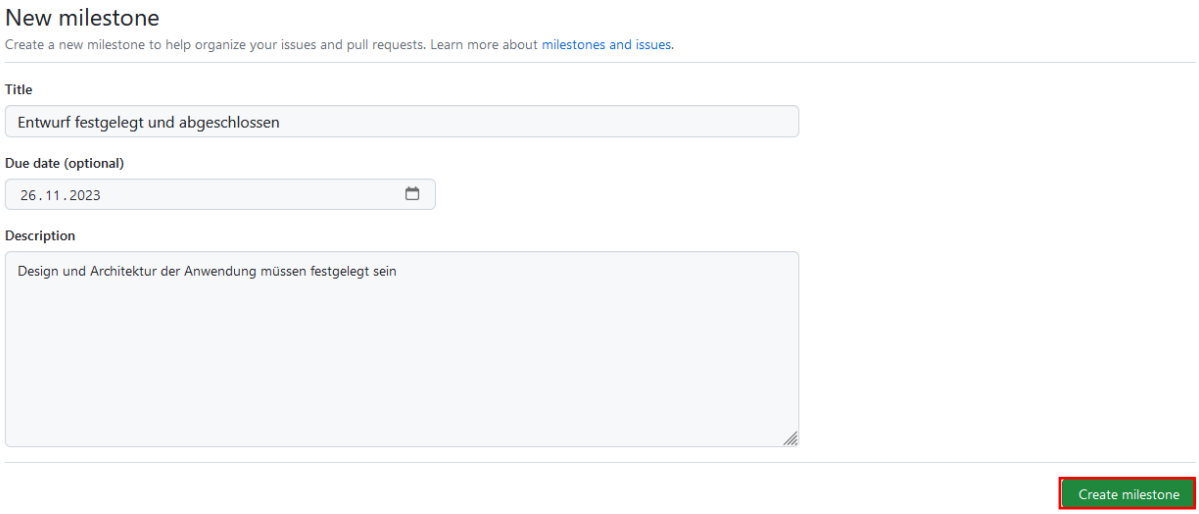


Initially, you roughly subdivide your project based on the milestones' results. Exemplary milestones could be planning completion, design completion or prototype development. Remember to give the milestones a completion date and a more detailed description. Create the milestones in consultation with your team. Each team member must create milestones. This is part of the later evaluation.

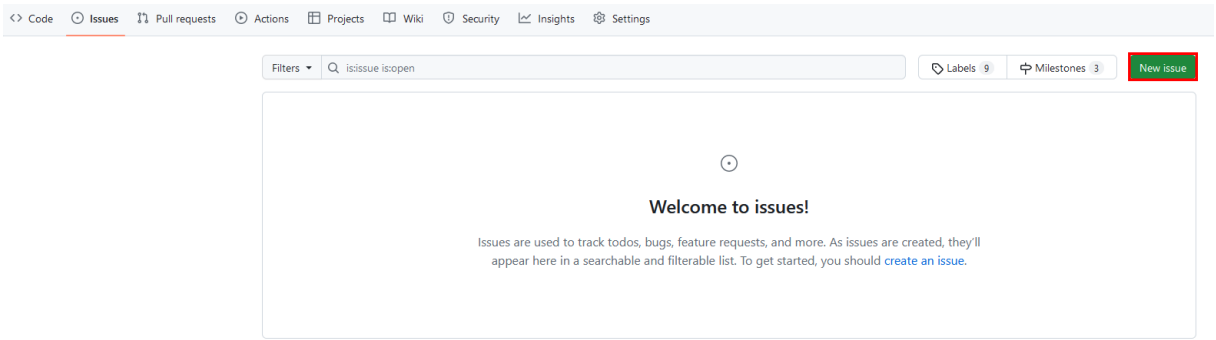
The following image shows how to create a new milestone. Create all milestones according to the planned project goals to be achieved in consultation with the team.



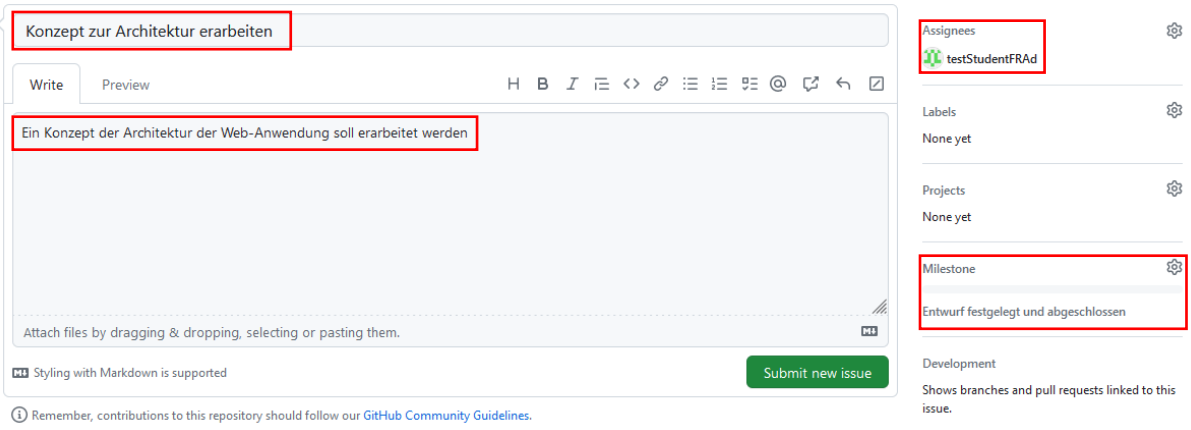
Name the milestones as meaningfully as possible. A designation such as Milestone 1 or First Step is not meaningful and, therefore not sufficient.



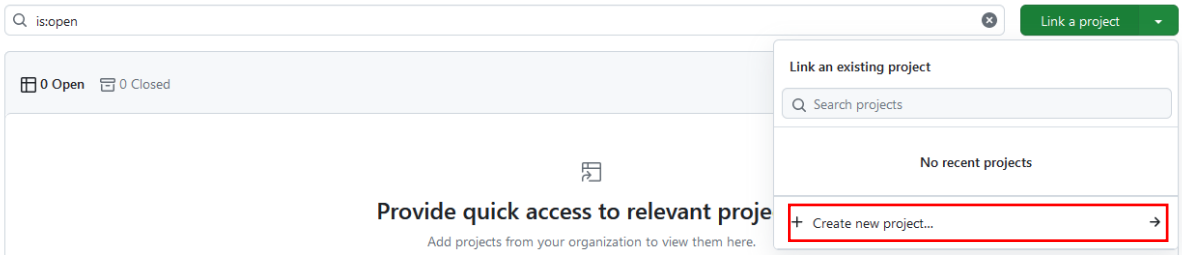
Once you have created the milestones as a rough plan for your project, you now create your work packages (Issues) for fine-grained project planning. Now, create these work packages using issues. Each team member must create work packages. This is part of the later evaluation.



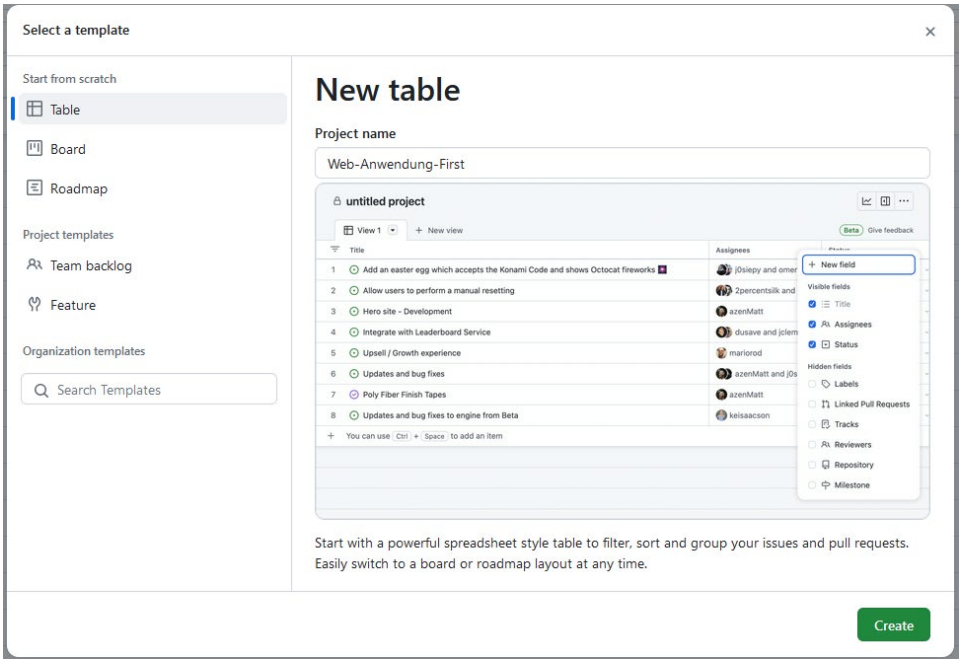
On the right-hand side, you define the assignees of the work package (see Assignees below). Under Labels, you can use an existing label or create your own. However, you can also adapt and change this during the project. What is important here is the assignment to a milestone that is to be achieved in part with the work package. Remember that milestones always consist of several work packages.



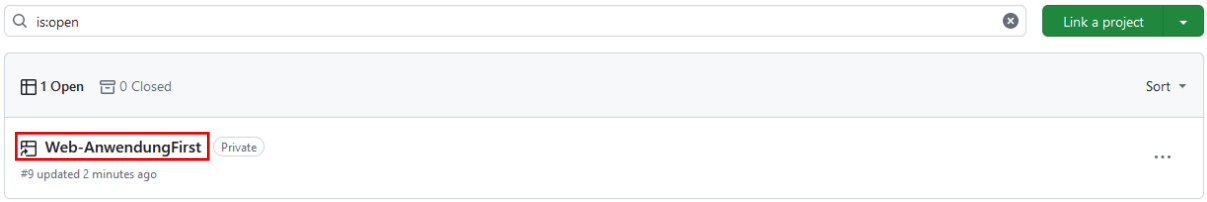
Once you have created several work packages for all your milestones, a project should now be created. Switch to Projects in the top tabs.



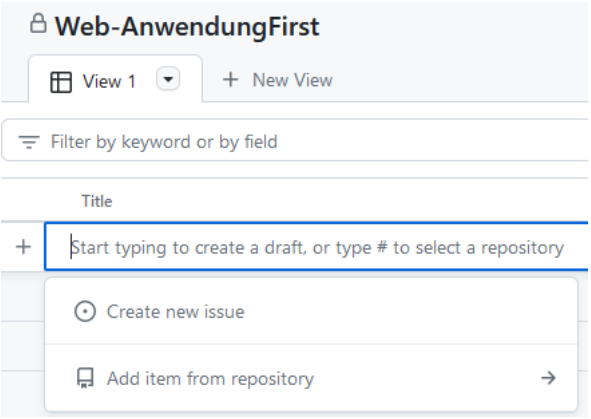
Create a new project and give it a meaningful name first, e.g. the name of your project. Leave the selection set to Table.



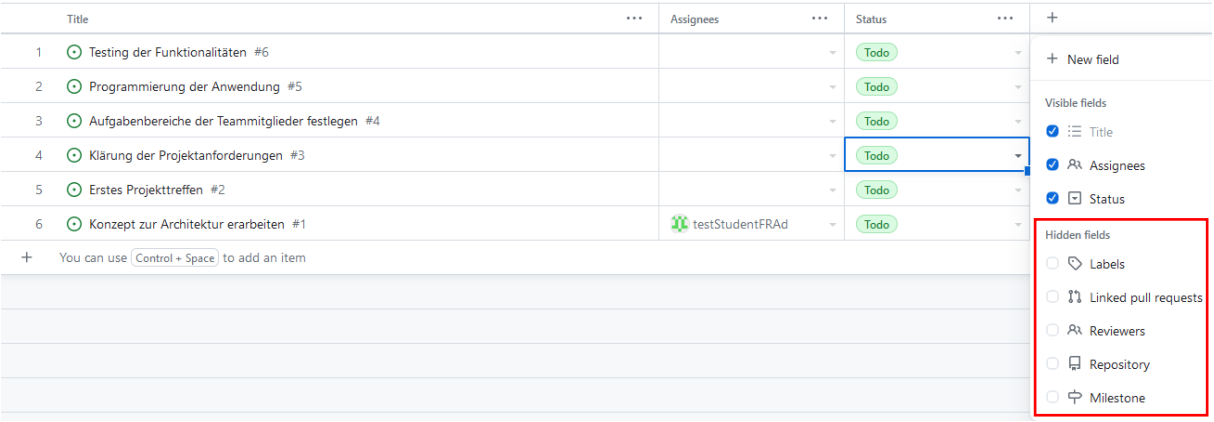
Then switch to the project view.



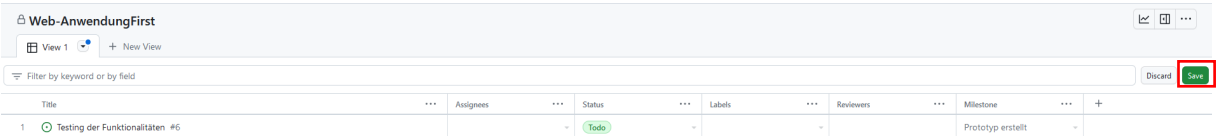
Here you can now add your previously created work packages by clicking on the + symbol under Title. And call up Add item from repository.



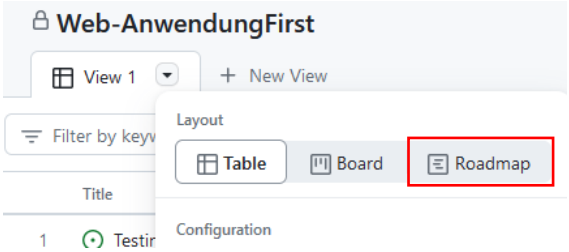
You will now see the individual work packages in the table view. Here you can assign the current status to the work packages, e.g. Todo, In Progress or Done. You can also define additional statuses if required. Use the + symbol in the top row of the table to add two more fields, Labels and Milestones.



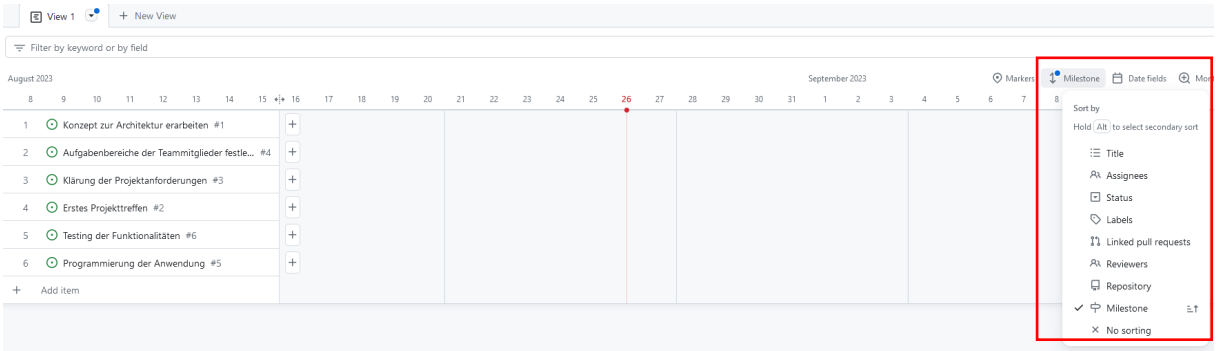
Next, save the table by clicking on Save.



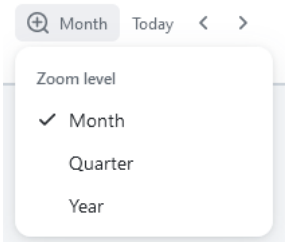
Then use the drop-down menu under View 1 to switch to the roadmap view. (Note: You can change the name of the view at any time!)



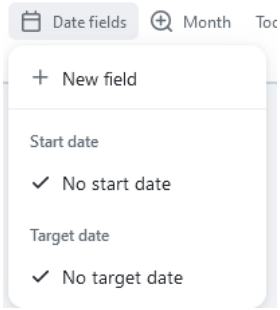
Here you should start sorting the work packages according to milestones.



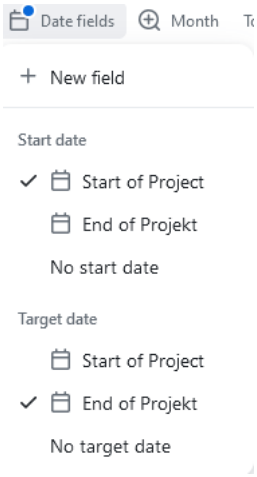
You should now define the timeline for the individual work packages. First, adjust the timeline by clicking on Month and selecting the desired time frame display, e.g. Quarter.



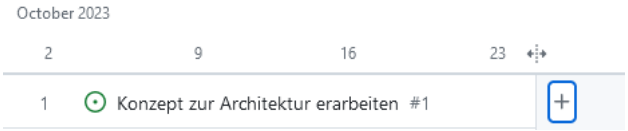
Now create two fields via the Date fields selection.



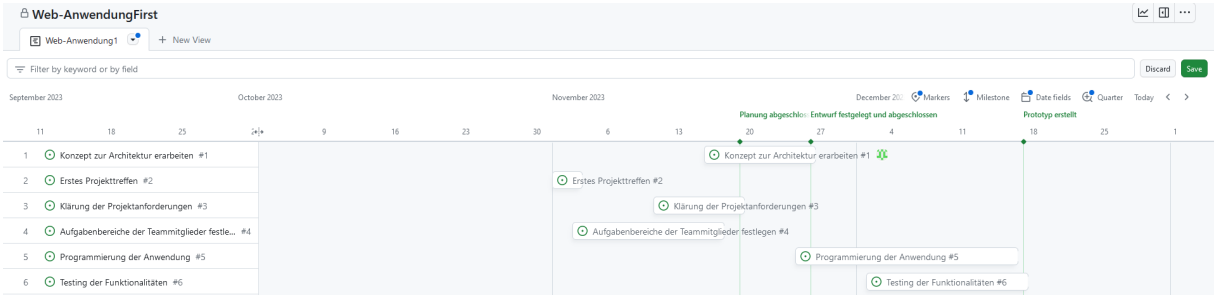
A field that defines the start of your project and a field that defines the end of your project needs to be defined.



Now select the respective + symbol in the line behind the first work package.



You can then arrange this on the timeline and define the duration of the work package by specifying the start and end dates. You can also display the milestones by selecting Markers. After the initial planning of your project, a roadmap overview such as this should have been created.



Always remember to keep the project overviews up to date as your project progresses. This includes, for example, adjusting the status of the work packages or adding new work packages.

Be sure to draw up a project plan at the start of your project. This is one of the foundations for project evaluation.

4 Project schedule & assessment

The project duration is approx. 10 weeks, excluding winter and Christmas vacations.

Project start: 12.12.2025

End of project: 06.03.2026 23:59 p.m.

Submissions must be made by no later than 06.03.2026 23:59 p.m.

In addition to the project planning in GitHub, a working implementation or configuration that meets the project description is required. A presentation of all team members involved in the application must also be created with Panopto or a similar application and submitted in CampUAS by the submission date.

In case of questions or problems please use the forum provided in CampUAS. In case of major difficulties, a meeting may be scheduled to explain the problem further. But please always try to solve the issues and research answers on your own first.

The project and the final evaluation are based on the following criteria:

I. Lecture (Presentation) (25%)
- comprehensibility of content
- structure and organization
- technical depth
- clarity and examples
- presentation and appearance
- use of media
- time management
- teamwork
- interaction with the audience
- overall impression

II. Project continuity (continuous working method) (15%)
- only recognizable continuous performance can be evaluated
- a realistic work plan is essential
- realistic structure of the workload distributed among the team
- individually assessable (Commits, Milestones, Issues, but no comments)
- regular work on project planning, documentation, implementation

III. Project planning (25%)
- continuity (see point 1), Milestones and, Issues (no comments)
- during the entire duration of the project

IV. Project Documentation and Implementation (25%)
- continuity (see point II), especially commits and their quality
- who contributed what and how often (commits and continuity)
- quality of the documentation
- complexity/volume
- project structure
- no branches other than the main branch are evaluated!

V. Project Presentation incl. Demo (10%)