# Project Software Engineering (DLMCSPSE01\_CF)

The goal of this project is to develop an AI based nutritional assistant. This assistant should help users to improve their diet and overall health.

The assistant targets people how struggle with healthy diets because they lack the right information or people already have the basic knowledge but need special advise in their daily life. Since this is a digital offer, the early adopters are digitally savvy and probably young people.

## Project plan

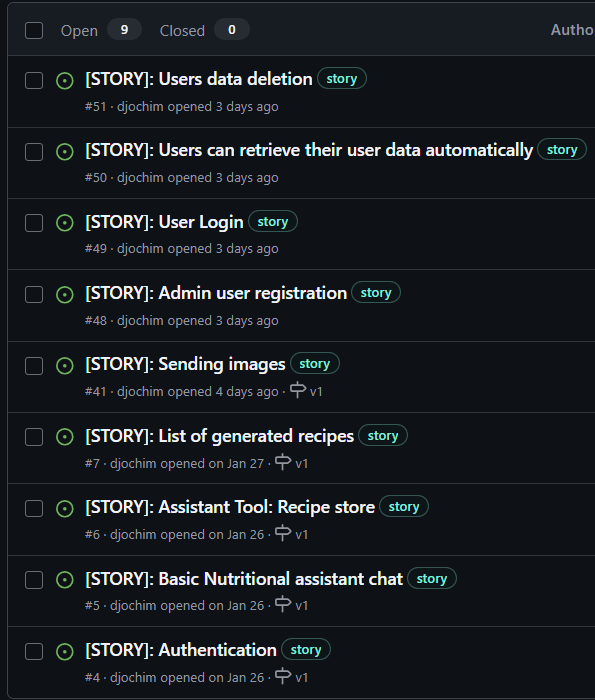
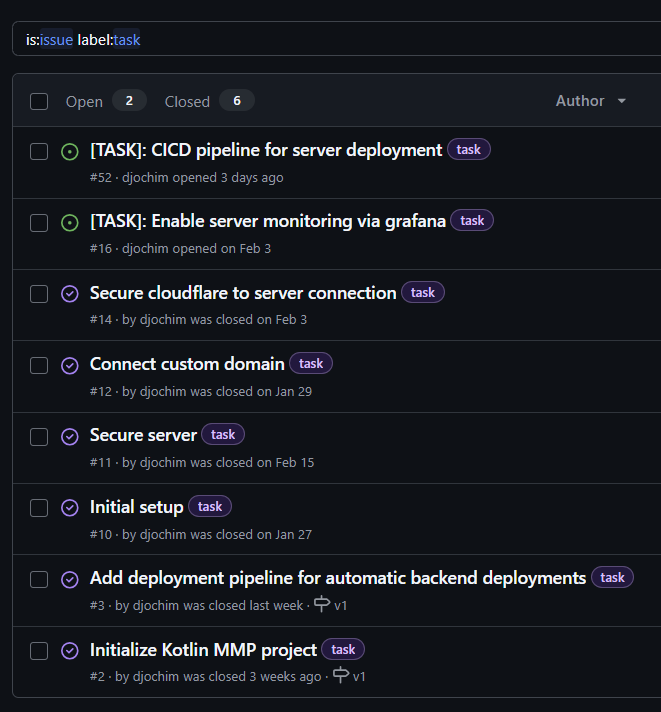
The scope of this project is restricted to milestone one. For the first milestone the app should only support basic chat functionalities between the user and the digital assistant. To demonstrate a first more advanced feature, the first assistant tool will be implemented. The first tool will allow the assistant to store recipes in the database. With this the first milestone will be implemented with those four epics.



The Epics are mostly independent from each other, but to simplify testing, the setup of the server and CI/CD pipelines should be completed before starting functional implementations.

The epic “User recipe store” should be implemented after the chat itself, because the chat is the only way, how recipes could be created. Nevertheless the retrieval of recipes could theoretically also be implemented before.

Technical functionalities are maintained as tasks. Those tasks are mainly part of the epic “Setup Cloud Server”. Feature enhancements are tracked via User Stories.



## Risks

A digital offer and the associated privacy concerns with those might scare off users. Especially because the assistant needs personal information about the user, to be able to support them with personalized advise.

As outlined before, the goal is to use an AI to answer the questions to the user. The output of those generative models can only be controlled in a limited fashion. The risk is, that the AI provide some negative advice to a user.

## Software development methodology

This project will use the Kanban software development methodology. Kanban is very lightweight which makes it suitable for this small one person project. It does not require complex planning and does not force some cycles, but still allows to be flexible and to change priorities.

### Kanban Board

The core focus in Kanban is the Kanban-Board. This project will be managed via Github-Projects and the Kanban board has the six columns Backlog, Refined, Ready, In progress, In review and Done.

The Backlog contains a list of all tasks for a bigger timeframe. It has a very big limit of 20, because in this area all items are collected for the future. Items in the Backlog are only roughly described and are not estimated and not prioritized.

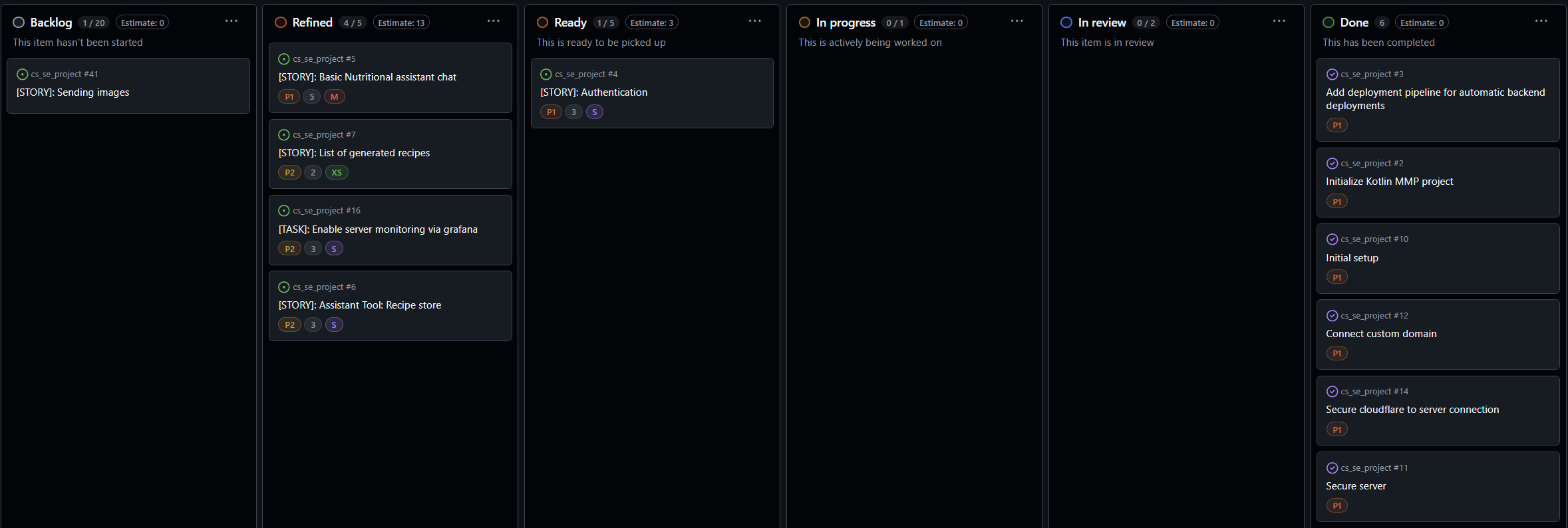
Items which are well described and which will be tackled in the next weeks, are in the Refined column. The column is limited to five prioritized and estimated items, which are ready for development as soon as all dependent requirements are fulfilled.

Once the refined items are completely unblocked, they can move in the ready column. These items could be started at any point in time. This column is also limited to five items and combined with the five refined items, they show the next up to ten items that will be done.

Started items are moved to in progress. Since this is a one person development project, the progress column is restricted to one item. With the multi-tasking will be prevented.

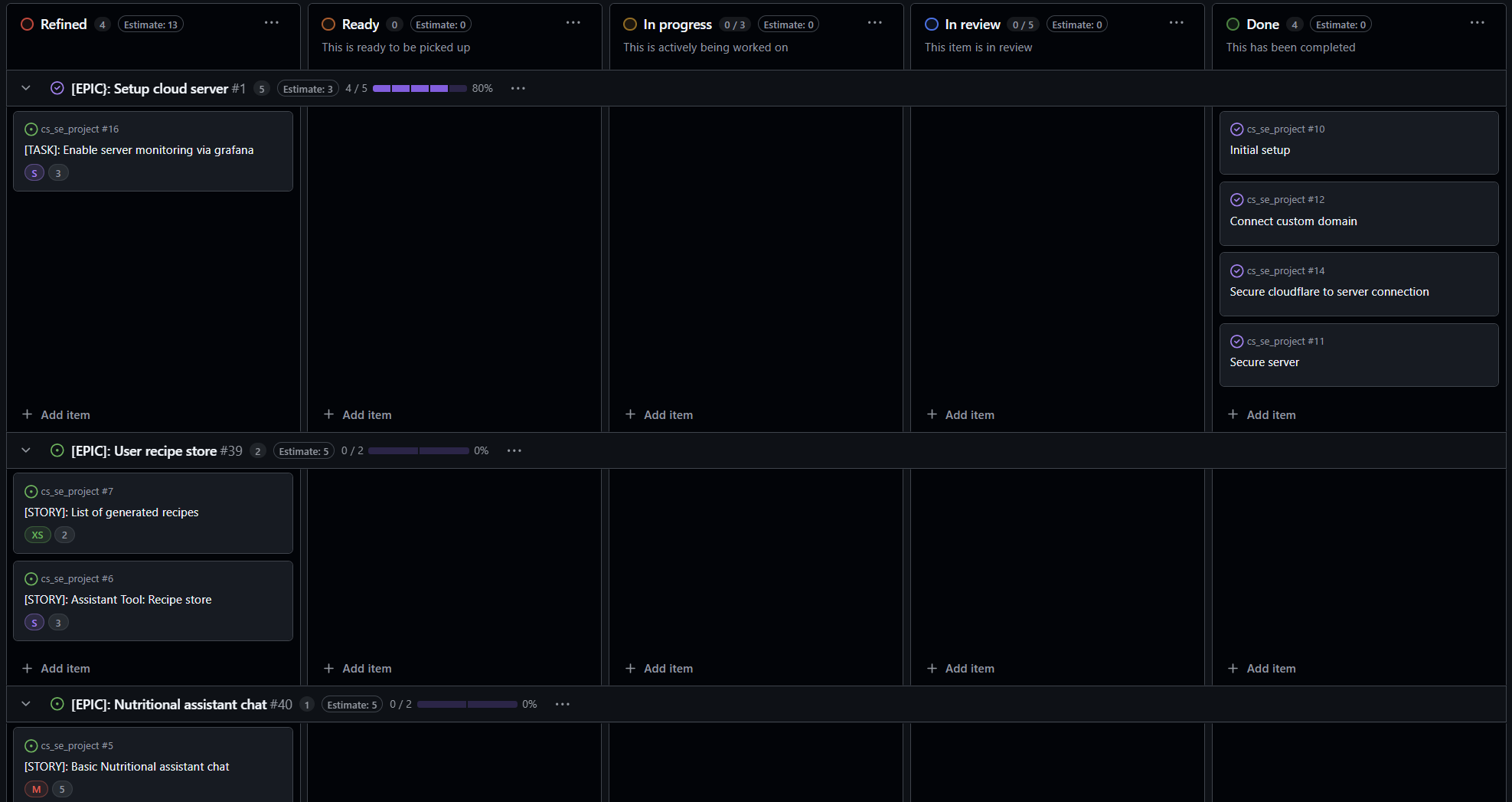
Once a PR is opened, items are automatically moved to In review. There are no second eyes, that could review the code, but there are automated checks, which prevent security breaches and software regressions.

After a PR is merged, items are moved to Done. The Done column has no limit configured.



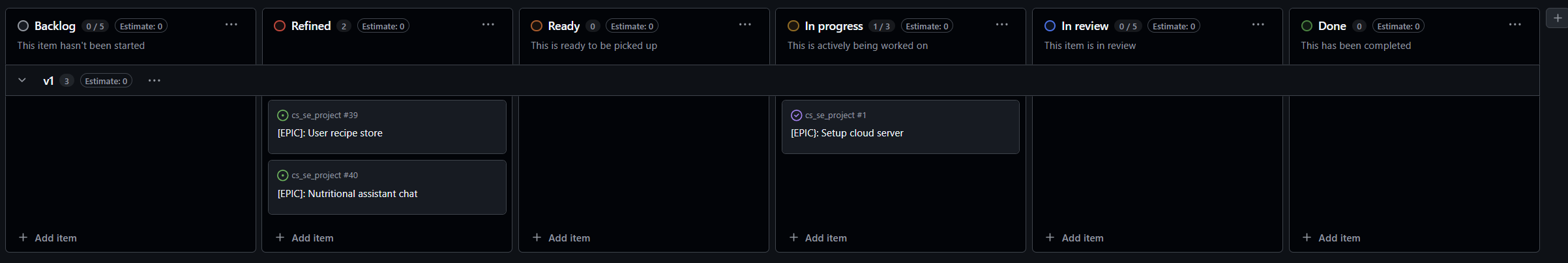
### Epic Board

The epic board shows a high level overview, how far the epics have progressed and which tasks/stories there are left to complete an epic.



### Milestone View

The milestone view show the status of the overall milestone and which epics are open until the milestone is completed.



### Roles

The Kanban project requires multiple roles which will all be filled with the on person in the exceptional project.

The first role is the Product Owner, who is responsible for creating the backlog and maintaining priorities based on user requirements. The Epics and User stories which the product owner defines, are further refined by the development team. This team consist of architects, developers and DevOps Engineers. The Architect is responsible for the system design and application architecture. The DevOps-Engineers create a continuous integration and continuous development pipeline. The pipeline ensures that new features are release often and only if the quality stages are passed.

Developers are responsible for the implementation of the tasks and stories. They are also responsible to write good/automated tests that ensure that new releases don’t introduce regressions an that the manual testing effort is low.

## Requirements

The requirements can be grouped into non-functional and functional requirements. The non-functional requirements are required to deliver a good software solutions with high quality.

### Non-functional

In alignment with a lean development process, a continuous integration and deployment pipeline is required for a continuously flow of new items to the user. A backend server must be stetted up as a proxy to the OpenAI assistant API, but also to implement additional functionalities

### Functional

For the first milestone the health coach has a very limited scope. First of all, the basic communication will be enabled between the Assistant and the user. This communication is based on text messages and the user can also send images.

In addition there will be a first toll implementation that allows the user to store recipes that are generated or extracted from text, images or websites.

## System Design

The application will be develop as Kotlin Multi-Platform project. There the server, Android App and iOS APP is develop with Kotlin and all three are also able to share some of the code parts with each other. The backend server uses Ktor as a server framework and connects to a MongoDB as storage.

The server will be deployed on a Hetzner Cloud Server as Docker container. For enhanced security the server will be behind a Ngix proxy and the whole Hetzner server will only be reachable over Cloudflare.

