User stories

US.01: As a user I want to choose a category of beers so I don't have to search the entire list.
US.02: As a user I want to find a beer based on my given criteria, so I don't have to search for a specific beer.
US.03: As a user I want a overview of the most popular beers, so I can see the best rated beers.
US.04: As a user I want to be able to post a commend under a beer, so everyone can see my opinion about that beer.
US.05: Als a user I want to be able to give a review at a beer, so everyone can see the score of a beer.

Requirements

Functional requirements:

FR.01: An user can read information about beers in an overview.
    R.01.01: An user can filter the list on category, alcohol percentage and price.
FR.02: An admin can add a beer to the overview
    R.02.01: The correct data is filled in: Name, Alcohol percentage, Category, Price and description.
    R.02.02: The price and alcohol percentage can't be negative.
FR.03: An admin can edit the data of a beer.
    R.03.01: The correct data is filled in: Name, Alcohol percentage, Category, Price and description.
FR.04: An admin can delete a beer from the overview.
FR.05: An visitor can create an account.
    R05.01: The email and password must meet the restrictions of auth0.
FR.06: An visitor can log in to their account.
FR.07: An user can post a comment under a beer.
    R.07.01: An user can only post a comment when the user is logged in.
FR.08: An user can give a review about a beer.
    R.08.01: The review has to be between 1 and 10 stars.
    R.08.02: An user can only give a review when the user is logged in.
FR.09: Users can chat with each other.
    r.09.01: Users can only chat with each other when they are both logged in.

Non-functional requirements

NF.01: The front and backend will be written in English so other developers can understand the code. (Maintainability)
NF.02: The frontend will be displayed in English so everyone who wants to use the application can use it. (Usability)

Moscow

With this Moscow method I describe what requirement Must be, Should be, Can be and Won't be in the application. The must haves have to be in the application. Without the must haves the application can't run properly. The should haves are very desirable but the application can run without it. The could haves are going to be implemented when there is enough time. The wont haves are not going to be in the application but can be interesting in a follow-up project.

Must haves:
FR.01, FR.02, FR.03, FR.04, FR.09
Should haves:
FR.05, FR.06
Could haves:
FR.07, FR.08
Won't haves: