SOA-Project: Personal Blog

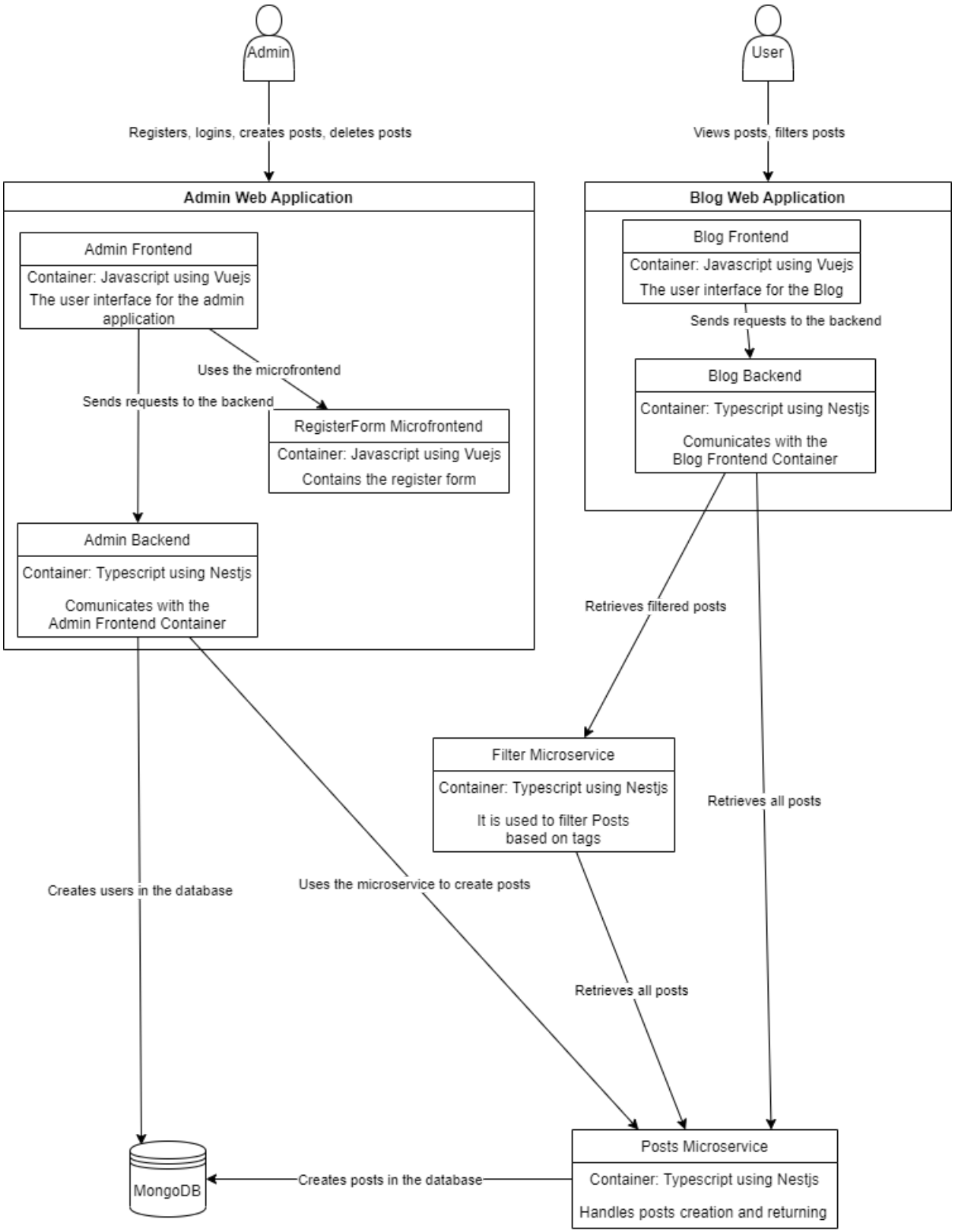
Thiess Cristian

This project is a personal blog split in two web applications. First we have an application which represents the admin part of the blog. Here user accounts can be created, users can log in and create and delete posts. These posts can be seen from the public part of the blog which is the second web application. On the public part users can just view the posts and also filter based on the tags that the post contains.

All the data is stored in a mongodb database.

The whole application is split into multiple processes which are running on Docker.
These are all the parts of the application:

- admin
- admin-frontend
- blog
- blog-frontend
- filter-microservice
- posts-microservice
- register-microfrontend


Admin
User
Registers, logins, creates posts, deletes posts
Views posts, filters posts
Admin Web Application
Admin Frontend
Container: Javascript using Vuejs
The user interface for the admin application
Uses the microfrontend
Sends requests to the backend
RegisterForm Microfrontend
Container: Javascript using Vuejs
Contains the register form
Admin Backend
Container: Typescript using Nestjs
Comunicates with the Admin Frontend Container
Blog Web Application
Blog Frontend
Container: Javascript using Vuejs
The user interface for the Blog
Sends requests to the backend
Blog Backend
Container: Typescript using Nestjs
Comunicates with the Blog Frontend Container
Retrieves filtered posts
Filter Microservice
Container: Typescript using Nestjs
It is used to filter Posts based on tags
Retrieves all posts
Creates users in the database
Uses the microservice to create posts
Retrieves all posts
MongoDB
Creates posts in the database
Posts Microservice
Container: Typescript using Nestjs
Handles posts creation and returning

1. Admin

This application represents the backend of the admin part of the app. It is written in javascript using the Nestjs framework. This app is using the posts microservice to submit new posts to the database and display the posts to the user.

This backend exposes these api links:

POST - /posts/

- This api call returns creates a post, the body of the call needs to contain the data for the post

GET - /posts/

- Returns all the posts

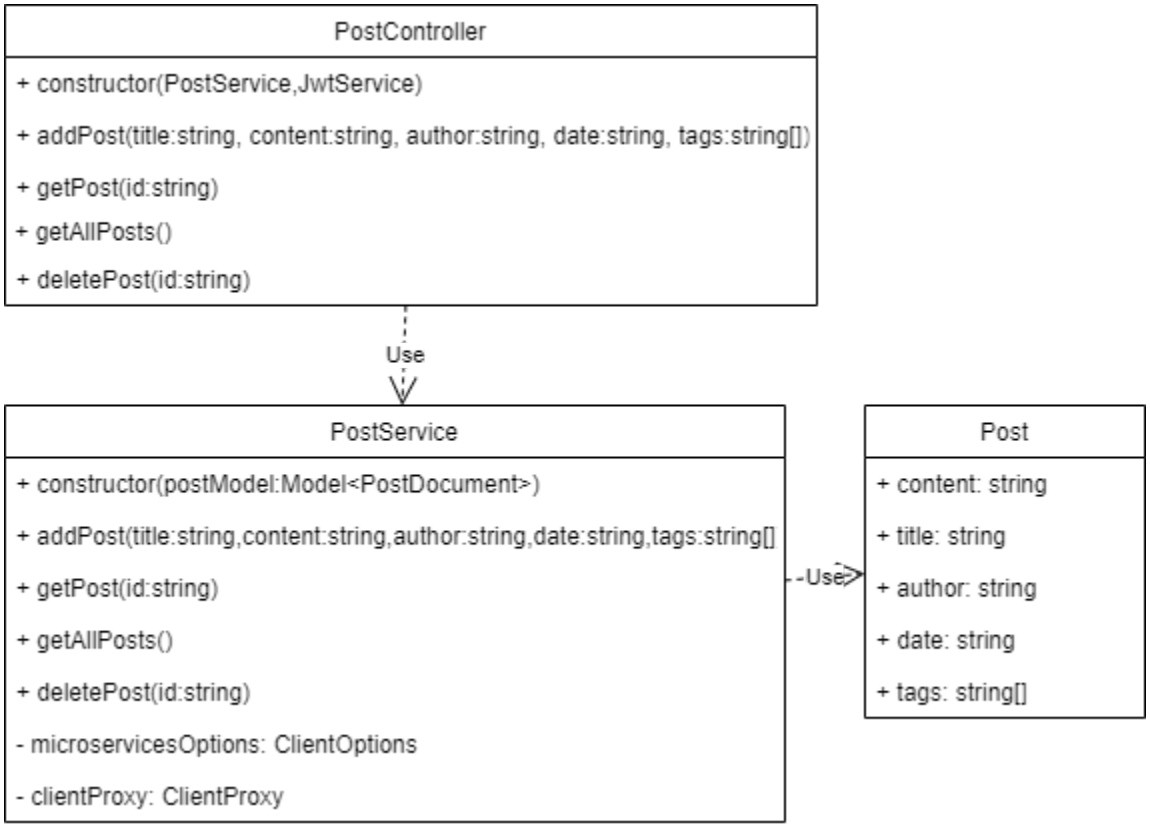
GET - /posts/{id}

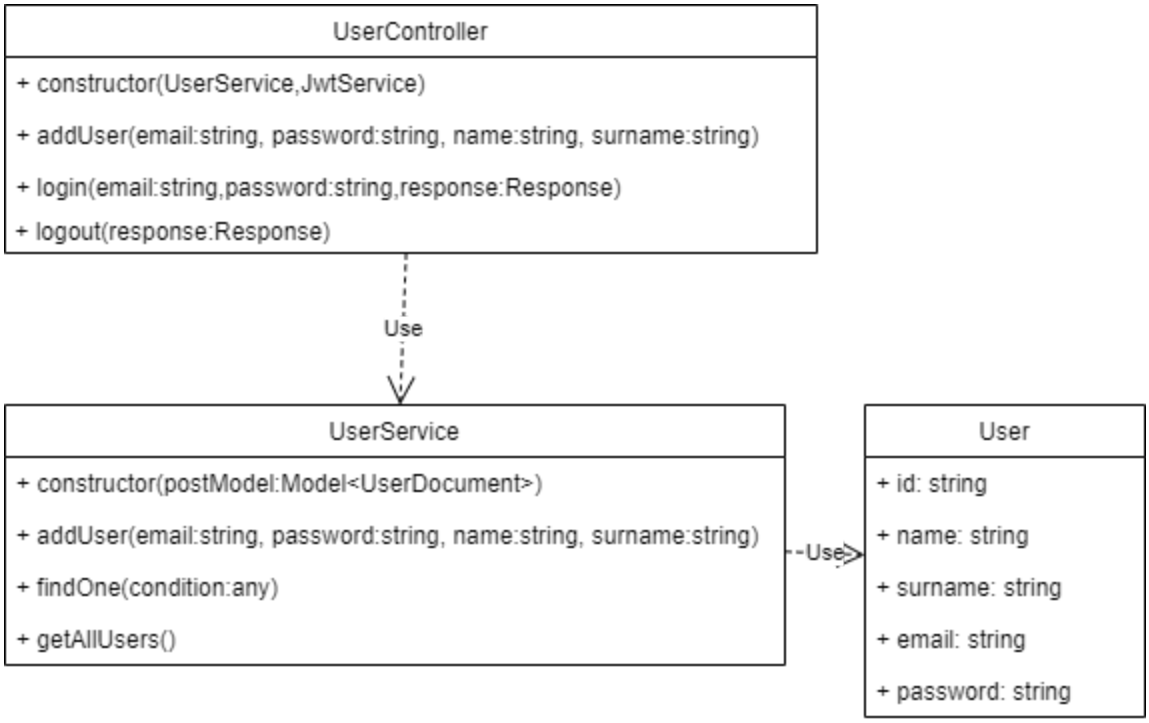
- Returns a post depending on the id

DELETE - /posts/{id}

- Deletes a post with the sent id

Bellow is a schema of the main classes in the application:



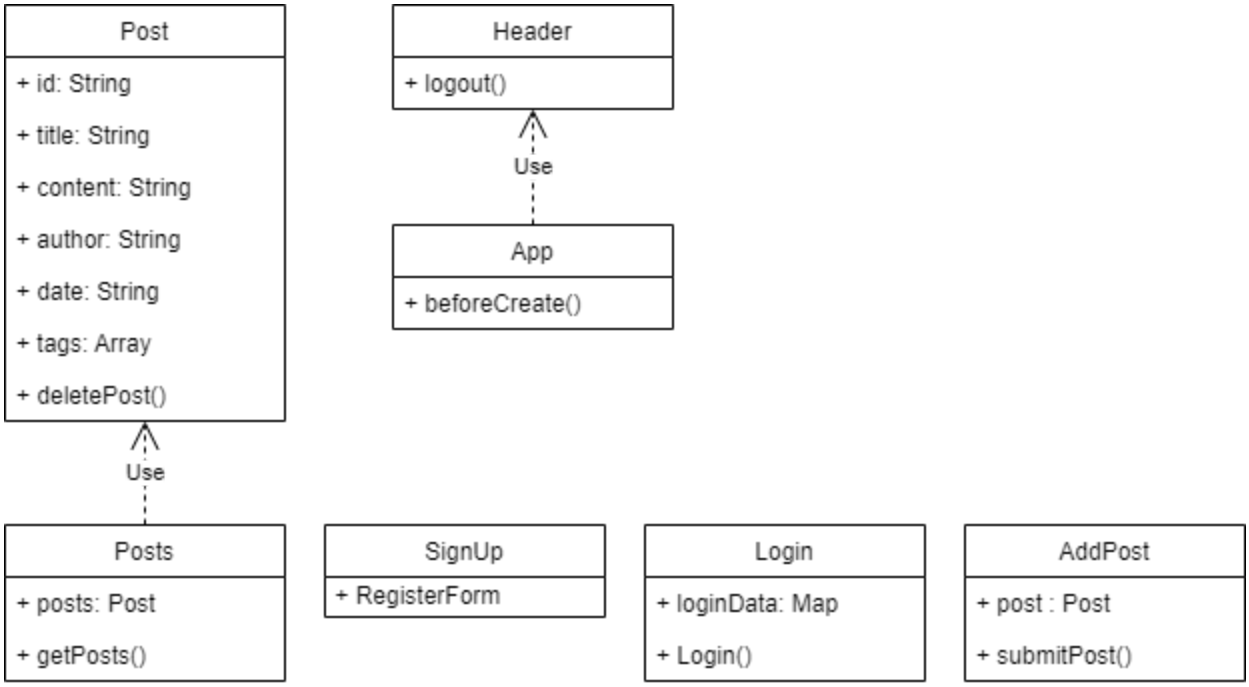


2. Admin-frontend

This application is the web client which enables the user to create and account, login, create blog posts and delete them. It is written using the javascript framework Vuejs. This web client communicates with backend app trough axios, which is a package that handles requests and responses to backend applications.

The frontend is built using components and views. A component is an entity which contains HTML, CSS and Javascript methods only needed there. A view represents a separate page.

Components: Post, Header
Views: Posts, SignUp, Login, AddPost



3. Blog

This application is the backend of the public blog application. It is also written in javascript using the Nestjs framework.

This backend exposes these api links:

GET - /filter/

- Returns all the tags that should be displayed in the frontend

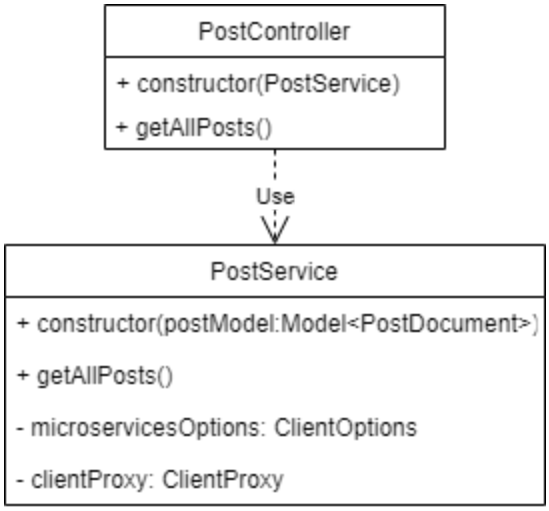
POST - /filter/

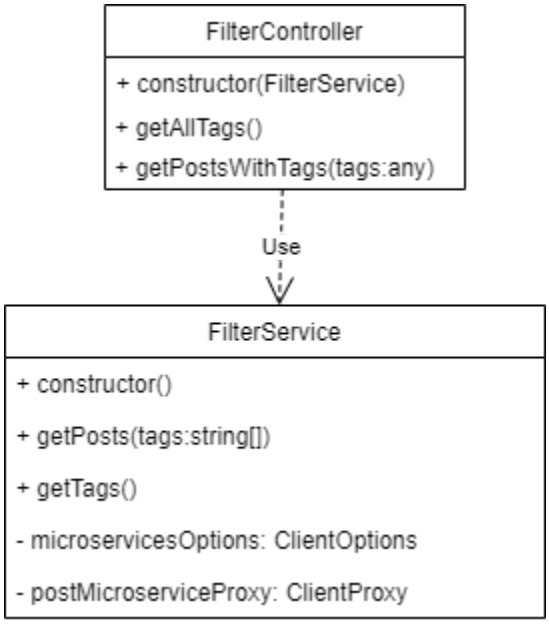
- Returns the posts containing certain tags. The tags need to be in the body for the request

GET - /posts/

- Returns all the posts

Bellow is a schema of the main classes in the application:

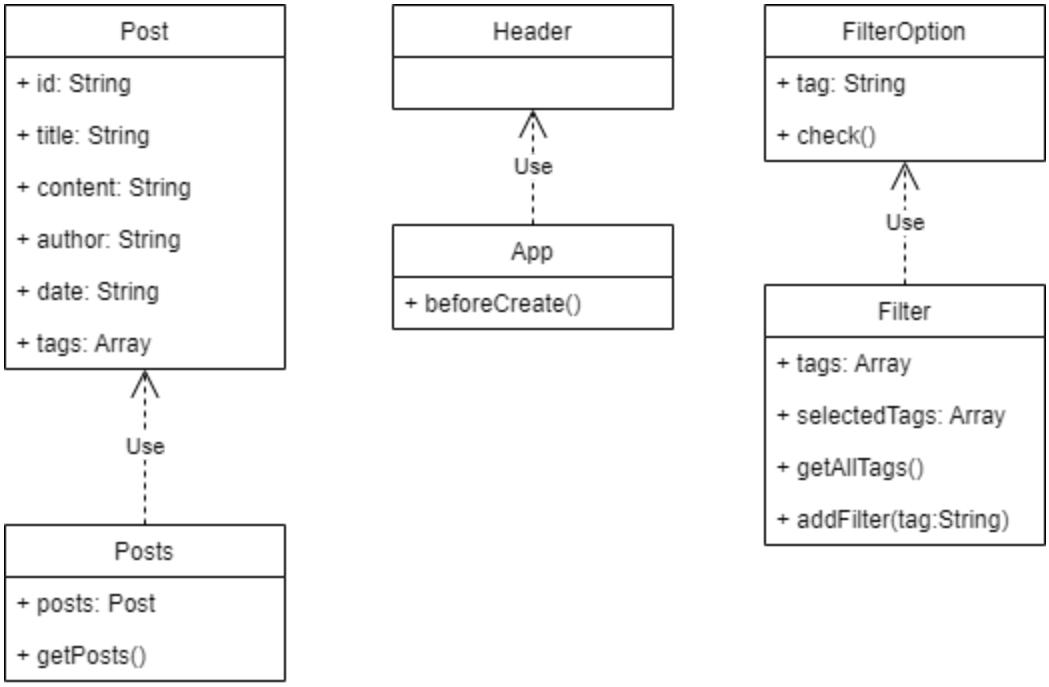




4. Blog-frontend

This application is the webclient for the public blog. It just displays all the posts. It also contains a filter which allows the user to filter posts based on the selected tags. It is also written in javascript using Vuejs.

Components: Post, Header, FilterOption, Filter
Views: Posts, SignUp, Login, AddPost



5. Filter-microservice

This is a microservice which is used to filter posts.

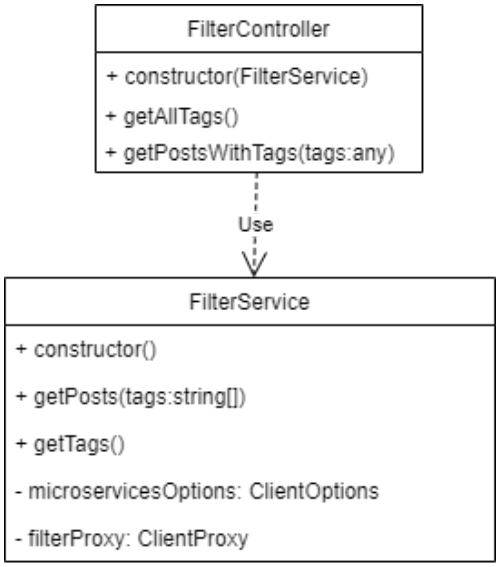
It has the following message patterns:

get_posts

- Needs a parameter which represents the selected tags. Returns all the posts containing those tags

get_tags

- Returns all the tags



6. Posts-microservice

This microservice handles the creation of blog posts.

It has the following message patterns:

add_post

- Adds a post in the database. The body should contain all the data for the post

get_post

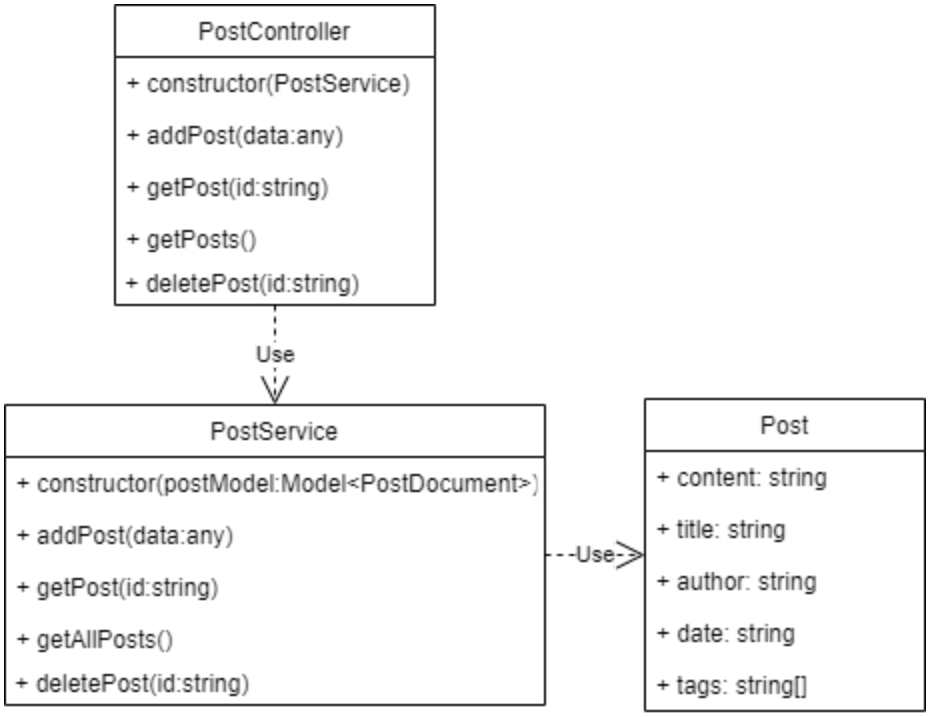
- Returns a post. It has a parameter corresponding to the id of that post

get_posts

- Returns all the posts

delete_post

- Deletes a post. It has a parameter corresponding to the id of that post



7. Register-microfrontend

This is a microfrontend app which exposes a register form which was used in the admin frontend application. It does this through webpack.

It exposes this Vuejs component:

RegisterForm