# Architecture

Disclaimer

This project will be my personal blog, which means that I’ll be the only one to create posts. I will not monetize this website.

## Project definition

Inspiration: [link](https://www.reddit.com/r/learnprogramming/comments/mumrgn/how_to_plan_and_build_a_programming_project_a/)

What is this project?

A personal blog that will have minimalistic design. Development will contain 3 stages:

* Stage 1 – MVP:

A read only blog with it’s design. It will have only home and about page. Home page will have posts sorted in chronological order. Inspiration: [link](https://oblac.rs/) and [link](https://herbertograca.com/).

I want to setup a monolithic architecture. Most of the frontend should be done in this stage, since it’s minimalistic.

Posts would be added either through database or through Postman. Frontend should expect this, and automatically format posts.

* Stage 2 – Viable product:

Should contain login for content creator, with authentication. Authorization won’t be necessary since there will be only one account that will have all permissions. User creation will be done through database, or through Postman. Login button will be hidden on the homepage *(this is debatable)*.

When logged in, user will be redirected to a content creation page. Also, there will an option to go to homepage which will have editable posts that are already posted.

* Stage 3 – Feature stage:

Having the core of the app intact, next thing to do is add features. Firstly, the reason why I decided to have a feature stage is that I want to practice adding features on fully functional app. Right now I have 2 features in mind:

1. Comments feature:
   * Comments v1 - *depricated*

Database setup for comments will be a takeaway from SQL Antipatterns book, which is a separate table for comments relations. I’ve wanted to try this for some time.

Comments feature will have oldest and newest sort.

Comments feature will have upvoting system *(debatable)*.

I am still not sure how I want to handle comments owner, are they going to be anonymous, or are the owners able to leave a signature that represents them, such as email or a nickname.

* + Comments v2

Inspiration: [link](https://xoofx.com/blog/2020/03/05/stark-melody-dotnet-sel4/) and [github](https://github.com/xoofx/xoofx.github.io/issues)

The idea is to open issues on github and use comments from issues. This removes a need for a database.

1. Email service:

I am still not sure if I wanna do this. I do not plan to advertise anything other than the blog itself. I do not know anything about email service setup so I will most likely look into this.

1. Any further ideas that I might have will be revisioned and added here.

## Tech stack

* Frontend – Angular
* Backend – .NET 6
* Database –
  + v1: EntityFramework SQLite for comments. Data will be saved in markdown files, as I will save only posts, and they will be implemented in frontend.
  + v2: Get comments from github issues
* Cloud – Azure
* Code repo – Github
* Kanban – Github
* Container – Docker

Any additional frameworks and libraries that I use in the project will be added here.

## Conclusion

I will use monolithic architecture as this will be a simple CRUD project. Inside of this project, Service with dependency injection *(might be overkill)* will be used to handle both business and user layers.