Project proposal

Link to the project's GitHub repo: https://github.com/tboth/rust-course-final-project/

Participating students:

Name	Discord handle
Tomáš Both	tommyboth#8244, name set to Tomas Both on the rs-101 server
Matej Lánik	Matej#8875
Pavol Hradský	Palo#3423

Project overview and main goals
The main goal of this project is to create a full stack web app for managing blog posts. This app will be a CRUD web app, with some custom business logic. By doing this project, we want to learn the basics of modern web development in Rust and want to get a general overview of the currently available tools - writing a REST API for the backend, managing SQL databases with ORM and creating a template-based frontend. We will try to use async programming in order to achieve bigger throughput and better scalability. We will also experiment with multi-factor authentication for our app.

Main goals (functional requirements for the app):

- Visitors will be able to read blog posts and choose blog posts from an aggregated list, similar to popular blogging platforms
- Visitors can filter blog posts based on predefined topic or author
- Editors will be able to login in, create accounts, add, edit and delete blog posts
- Blog posts will contain a title, text divided into sections, and pictures (pictures will not be mandatory to use in blog posts, but implementation for providing images will be a part of the application)
- There will be an experimental option for multi-factor authentication for logging in
- All user data and blog posts will be stored in an SQL DBMS

Public crates we are planning to use

- **Rocket framework** - backend - https://crates.io/crates/rocket/0.5.0-rc.2 - v0.5, since this version provides async mode
- **Templates in Rocket** for frontend - https://api.rocket.rs/v0.4/rocket_contrib/templates/index.html - the main focus will be on the backend and database system, this is why a simpler fronend solution was chosen. We want to keep this part simple (while providing necessary functionality) and focus on the parts which will be written in Rust - the ORM and backend
- **Rocket session store** - authentication - https://docs.rs/rocket-session-store/latest/rocket_session_store/
- **SeaORM -** ORM with async support - https://www.sea-ql.org/SeaORM/
- **Oauth2** - https://docs.rs/oauth2/4.0.0-alpha.1/oauth2/index.html - for implementing MFA

Basic architecture diagram