# DISCUSSÃO

# Criação do Projeto

Na fase de planejamento, a primeira etapa do ciclo de vida de desenvolvimento de um software, foram definidos a topologia de implantação de microsserviços, a tecnologia a ser utilizada, suas instâncias, o método de comunicação entre as instâncias e os tipos de bancos de dados.

# Definir os microsserviços em grupo;

Foram levantadas todas as entidades pertencentes ao projeto, foi utilizado o software online <https://app.diagrams.net/> para a elaboração dos casos de uso, construção dos relacionamentos entre tabelas e diagramas de estado. . Utilizou-se da linguagem natural e obíqua para a compreensão de todos integrantes.

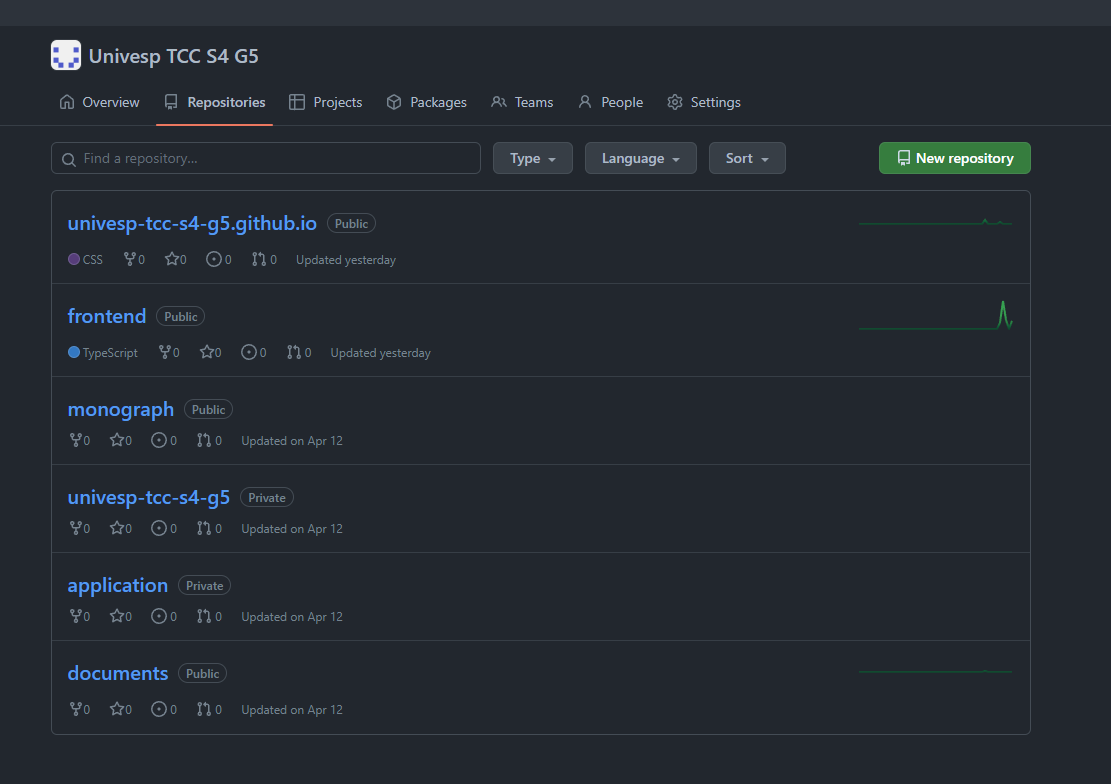
* 1. **Elaboração das telas**

Para se ter uma visão geral da quantidade de telas, layout e estética que a aplicação poderia ter, foram prototipadas algumas telas ulilizando a ferramenta Figma <https://www.figma.com>. Neste aplicativo se fez possível os primeiros testes de navegação e poder enxergar como a o aplicativo que estava sendo desenvolvido se portaria em dispositivos móveis, tais como smartphones e tablets, com a vantagem de ter sido possível compartilhar o trabalho em tempo real com todos os membros do grupo e testar a interface nos próprios dispositivos.

* 1. Criação do repositório

Para se desenvolver um trabalho em grupo e que todos pudessem receber as últimas modificações do que havia sido feito, foi criado um repositório no Github <https://github.com> utilizando uma conta do tipo "Organization", na qual é possível que vários membros possam ser convidados e utilizar a ferramenta com os mesmos níveis de permissão <https://github.com/univesp-tcc-s4-g5>. Para cada subtarefa foi criado um repositorio específico, sendo eles:

* Hospedagem do frontend no github pages: <https://github.com/univesp-tcc-s4-g5/univesp-tcc-s4-g5.github.io>;
* Codigo referente ao frontend propriamente dito: <https://github.com/univesp-tcc-s4-g5/frontend>;
* Codigo dos microservicos e dockerfiles: <https://github.com/univesp-tcc-s4-g5/application>;
* Gerenciamento das versões da monografia: <https://github.com/univesp-tcc-s4-g5/monograph;
* Documentação geral: <https://github.com/univesp-tcc-s4-g5/documents>;



Repositorio do projeto

Fonte: autoria propria

* 1. Desenvolvimento do aplicativo

A primeira etapa do desenvolvimento, foi definir a stack de programação, para isso foi decidido utilizar o MySQL como banco de dados, porque além de ser um dos mais conhecidos e difundidos no mercado, possui código aberto, o que possibilita o desenvolvimento sem adição de outros custos. Para a tecnologia de backend foi definida a linguagem Java, por também ser open source, multiplataforma e por também ter sido utilizada amplamente ao longo da graduação e a escolha desta tecnologia não impede que outros microserviços sejam criados, pois, utilizando esta arquitetura é possível integrar quaisquer outros serviços independentemente de sua tecnologia. Para a parte do frontend, foi escolhida a stack web, por permitir acesso universal de qualquer dispositivo, independentemente de seu fabricante ou sistema operacional. Para a orquestração dos microserviços, utilizamos o docker, pois atualmente é o maior gerenciador de containers disponível.

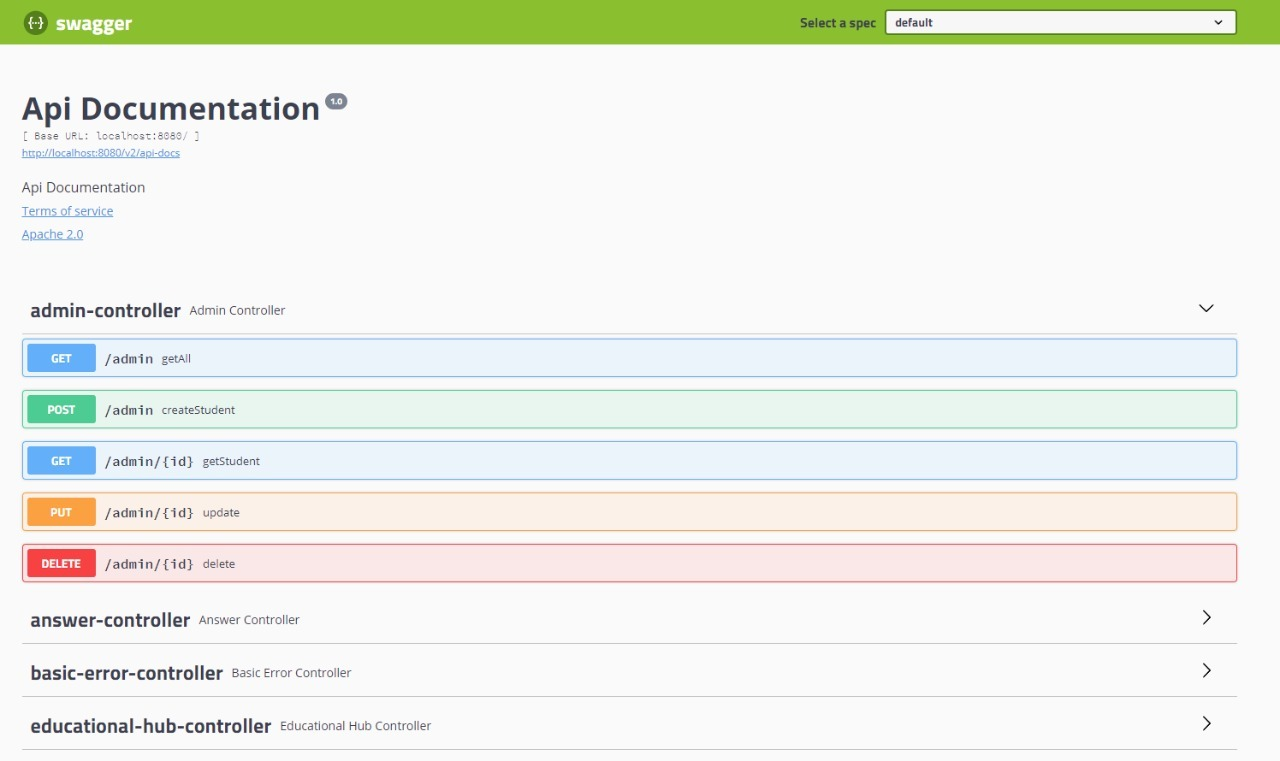
Para o desenvolvimento do aplicativo, foi necessário o download e instalação dos seguintes softwares:

* Git for Windows - gerenciamento das versões;
* Docker Desktop - permite que instâncias do docker sejam executadas na própria máquina do desenvolvedor;
* Visual Studio Code - para edição de códigos em qualquer linguagem de programação;
* IntelliJ Idea - para o desenvolvimento dos microserviços utilizando a linguagem Java;
* MySQL Workbench - para a visualização dos dados armazenados pelos microserviços;

Após tudo ser devidamente instalado, foram criados os primeiros microsserviços básicos, sendo eles: banco de dados e load balancer. Para realizar este passo, necessita-se criar um arquivo "docker-compose.yml", onde para cada secção é definido um serviço e seus atributos, os mais comuns são:

* Port - portas que serão expostas externamente
* Volume - endereço para onde o serviço salvará seus dados no disco do host
* Env - variáveis de ambiente passadas para configuração do serviço
* Nome - nome que aparecerá no gerenciador
* Host - endereço que o serviço irá responder internamente

Criados e configurados os primeiros serviços básicos, iniciou-se o desenvolvimento do backend, para isso, utilizamos o template do framework Spring Boot. Com ele foi possível configurar a conexão com o banco de dados, posteriormente as entidades que, após definidas, gerariam as tabelas no banco, as suas controllers para receber as requisições HTTP de acordo com a rota que o usuário solicitar, seus serviços internos e por fim, com todo a regra de negócio aplicada, gerar uma rota para a documentação automaticamente, isso só se faz possível pelo uso da biblioteca "Swagger", que gera todas as interfaces criadas na aplicação e disponibiliza em uma página do próprio projeto.



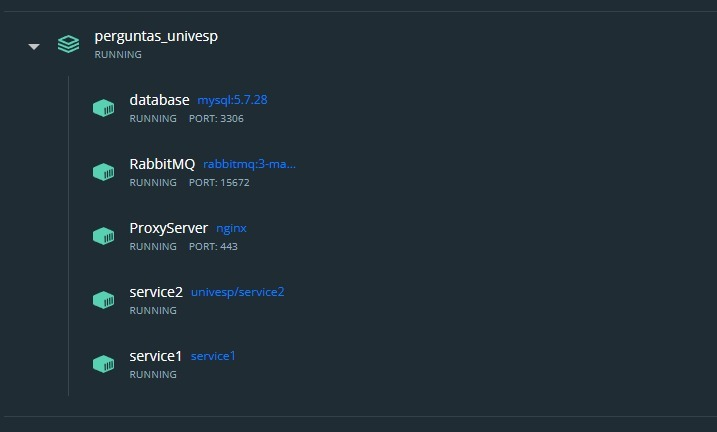
Documentação dos serviços

Fonte: autoria propria

Para o desenvolvimento do frontend foi utilizado o framework Angular, que possui um template muito simples para a criação rápida de páginas estáticas. Foi se criando as telas uma por uma de acordo com o layout estabelecido no Figma, utilizando a metodologia "Mobile First", na qual o desenvolvedor prioriza que a aplicação fique fluída com priorização nos dispositivos móveis e só depois melhorar a experiência para os dispositivos desktop.

* 1. Criação de microsserviços com o Docker

Com todo desenvolvimento finalizado, criou-se no mesmo arquivo docker-compose.yml uma configuração para cada serviço, após isto, iniciou-se os testes com todos os serviços para que fosse assegurada a comunicação entre eles. Também foram executados testes de consitência dos dados para cada serviço criado.

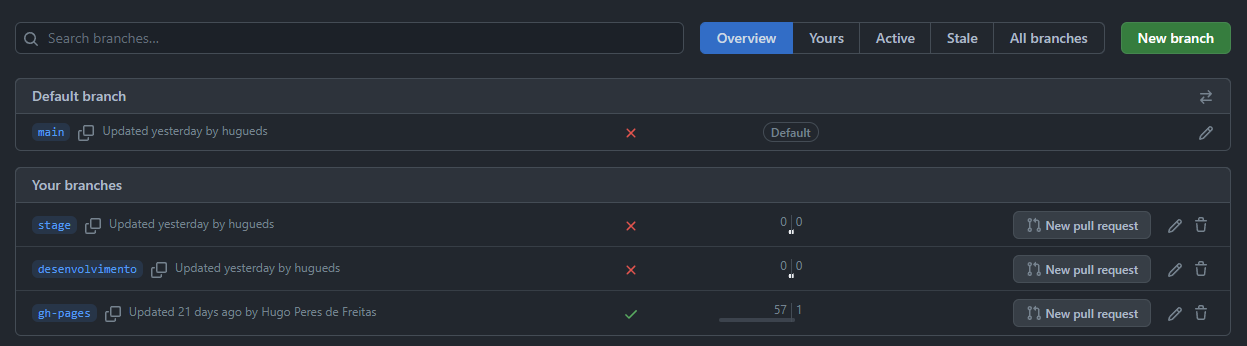


Serviços rodando com o Docker

Fonte: autoria propria

* 1. Criação de branches builds/ ou commits com o Git

Para hospedar todo o código e documentação criada de maneira organizada, foram criadas as branches de desenvolvimento, stage e principal, para cada projeto, sendo a principal (main) onde reside a ultima versão oficial para todos terem acesso. O acesso da branch principal é restrito apenas para leitura, sendo sua escrita feita pelas branches de desenvolvimento e stage automaticamente.

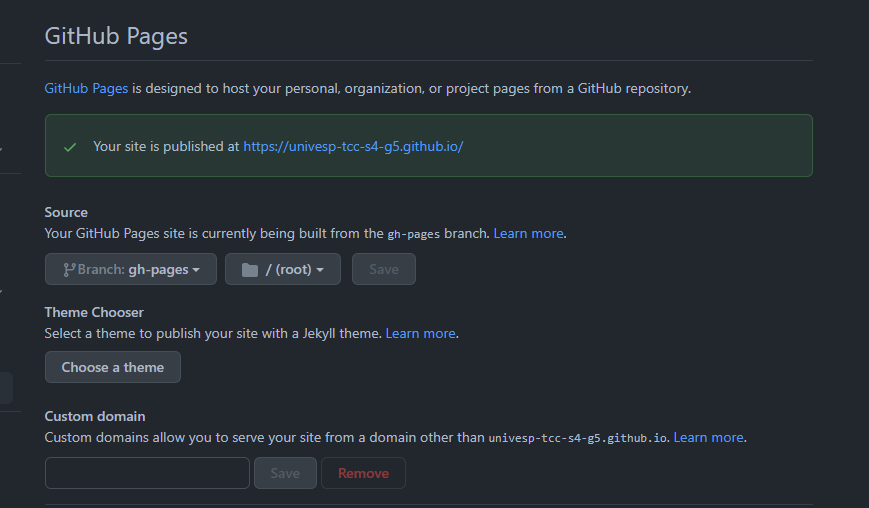


Branches do projeto

Fonte: autoria propria

* 1. Deploy

Com todo o código hospedado na plataforma, criou-se um outro repositório, este com intuíto de disponibilizar a página de maneira automatizada e gratuita com o github pages, todas as vezes que uma nova release é publicada na branch principal, um runner interno do github publica a nova versão do site em seu domínio.



Configuração das branches do projeto

Fonte: autoria propria