Esse é o documento feito pelo primeiro grupo do projeto (2024.1) e esses registros são da primeira reunião com o cliente.

final da primeira sprint - 19/03

reunião 28/02 - primeira reunião

mvp:(produto mínimo em que o usuário consegue testar coisas básicas no sistema) - ex: ver se os estados da aplicação e cadastros estão funcionando.

Finalidade do projeto:

Unimpact é um projeto que a universidade possui para promover a conexão entre comunidade e meio acadêmico, com o intuito de desenvolver projetos de extensão e pesquisas. Atualmente, todo esse processo de conexão e validação de propostas é feito de forma manual, com a ausência de plataforma, o que leva muito tempo, sendo válido automatizar esse processo.

-mostrar o projeto para a pessoa da coordenação de extensão para ele validar o sistema.

temos 3 agentes no sistema:

Usuário solicitante: (pessoas de fora da universidade - como empresas, ONG's) que querem fazer parceria com a unicap. obs: para isso precisam estar cadastrados no sistema

Usuário intermediador(é um funcionário da coordenação de extensão que verifica a proposta enviada pelo usuário solicitante e valida ou não essa proposta)

Usuário Executor(pode ser um professor ou qualquer funcionário competente para executar esse projeto em questão)

banco de dados usados: PostgreSQL

Melhorias:

- testes unitários para validar o backend, como testes de api(por exemplo, pelo swagger). mas aparentemente o backend está ok assim como os testes com selenium(teste automatizado para fazer o fluxo do usuário)

- validação e testes no banco de dados

- front-end estava muito caótico e a sugestão foi refazer do zero, e até mesmo modificar telas no figma (fazer com bootstrap devido a pessoas no grupo possuírem familiaridade)

- status da proposta do projeto(se foi aceito, se está em análise ou foi recusado. isso vai dar dinâmica ao projeto). isso precisa de testes unitários para verificar o estado do projeto

- manter o usuário logado independente de atualizar a página ou transitar entre páginas

- botões para mudança de estado (como botão de aceitar, solicitar ajustes, recusar, não tá funcionado, pois a api não está sendo chamada lá no front)

teste para verificar se após mandar uma requisição a api está devolvendo correto.

- tem bugs como ao clicar em "seus dados" ele redireciona para uma página em branco e clicar em voltar redireciona para uma página em branco.

M-1: Quando um dados for excluído do banco de dados, os metadados podem continuar armazenados, como uma forma de histórico ou análise.

Exemplo: A UNICAP quer saber quantos projetos foram realizadas até o momento, mas, (por algum motivo) um Usuário solicitante e dono de seus projetos, excluiu alguns da sua lista.. os metadados desse projeto podem permanecer no banco, e o status de exclusão é transparente para a UNICAP.

M-2: Uma lixeira de dados excluídos. Exemplo: Projetos excluídos podem ficar em uma lixeira por um período de tempo e depois podem ser excluídos definitivamente, caso não sejam recuperados pelo seu proprietário.

M-3: Não passar os parâmetros na URL da requisição para os próximos endpoints a serem criados.

M-4: Para que eu possa colocar (pendente) quando um usuário ainda não aceitou um convite, preciso que a notificação tenha a que grupo/organização o usuário está sendo convidado, caso contrário pode vir diversas solicitações que estão pendentes em grupos diferentes etc..

Rodar aplicação:

o docker é um container, e é uma forma de fazer a aplicação ser multiplataforma. no container teria todas as dependências necessárias.

o docker é um container que dentro dele tem uma imagem da sua aplicação

arquivos no repositório:

docker_compose.yaml -> nesse arquivo cria as dependências da view, da api

(instalar o docker e docker compose. obs: o docker desktop é uma interface gráfica e lá consegue startar todos os containers de uma só vez)

usar o swagger para verificar se o banco está conectado. caso esteja ok, caso não esteja, tem que fazer esses passos:

Apos clonar altere as seguintes informações no application-dev.properties

spring.datasource.url (nome do banco de dados e porta utilizada)

spring.datasource.username (usuário do banco de dados)

spring.datasource.password (senha do banco de dados)

-> o maven é o gerenciador de projetos do java assim como o gradle

na parte de gestão de projeto:

- determinar o que faremos até o fim do semestre