PROJETO DB

Grupo 02:

Nomes	Contribuição (CRediT)
Anita Garcia Lagos	Conceitualização, Revisão do Banco de Dados, Criação de Tabelas e Seeds no banco de dados
Caetano Amorim	Conceitualização, Desenho do Banco de Dados, Criação de Tabelas e Seeds no banco de dados
Gabriel Cardoso	Conceitualização, Revisão do Banco de Dados, Criação de Tabelas e Seeds no banco de dados
Thiago Carrijo	Conceitualização, Desenho do Banco de Dados, Criação de Tabelas e Seeds no banco de dados
Victor André	Conceitualização, Escrita - Rascunho Original, Desenho do Banco de Dados, Criação de validações no banco de dados e revisão das tabelas, Criação das funções de API

Configuração de ambiente:

Para a realização desse projeto vamos trabalhar no Linux e utilizar as seguintes tecnologias:

O PostgreSQL será nosso SGBD, mas não precisamos instalar ele diretamente porque vamos utilizar o **Docker** para instalá-lô. Basta seguir o tutorial correspondente a sua distro no linux na sua página oficial, porém no windows o link é esse aqui.

Também será necessário instalar o **Docker Compose,** basta seguir esse link.

Ambos serão usados para rodas o banco de dados em um container, facilitando a reprodução e o desenvolvimento do projeto.

O Python será nossa linguagem de programação principal, o qual será usado para fazer a api para se comunicar com o banco de dados e também para fazer o front end da aplicação.

Provavelmente, o Python já estará instalado no seu Linux, mas se não estiver basta rodar o comando: sudo apt install python3 se estiver utilizando o Ubuntu, caso contrário basta seguir o tutorial correspondente ao sistema aqui.

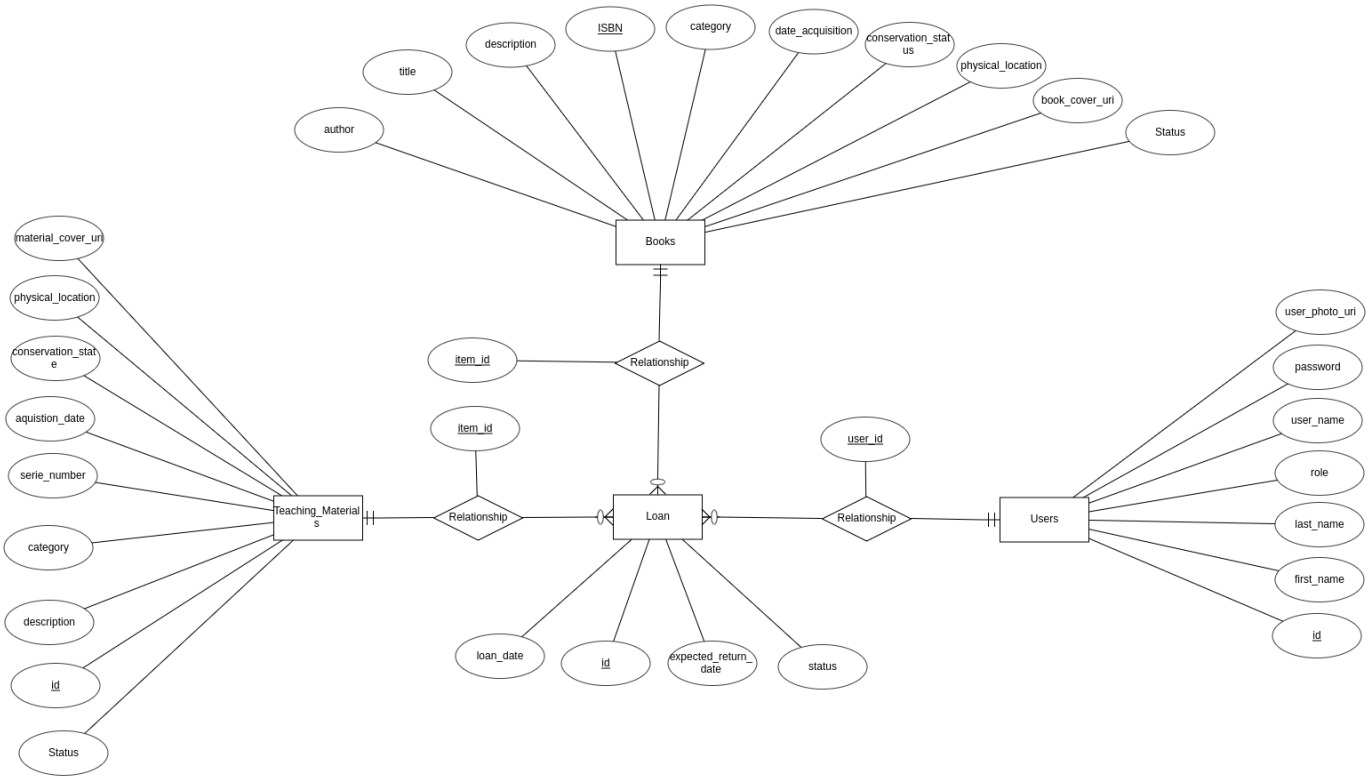
Juntamente com o Python vamos utilizar algumas bibliotecas, são elas:

- **Streamlit** - Para criar a interface gráfica
- **Psycopg2** - Para se comunicar com o banco de dados PostgreSQL
- **Passlib -** Para validação da senha

Para instalar qualquer uma delas basta ter o Python instalado e usar o comando pip install <nome da lib> no terminal que ela será instalada. Na versão final do projeto haverá um arquivo “requirements.tx” para instalar todas as bibliotecas em apenas um comando.

Modelos do banco de dados:

A seguir, temos o diagrama de entidade e relacionamento do banco de dados:



Aqui está descrito as 4 tabelas que serão criadas (Users, Books, Teaching_Materials, Loan).

A tabela Users existe para termos os administradores que vão criar os demais usuários, cadastrar os livros, materiais e fazer os empréstimos para os membros do laboratório, que são os demais usuários da plataforma. Aqui, armazenaremos os dados do usuário e suas permissões. Ela possui um trigger para encriptar a senha.

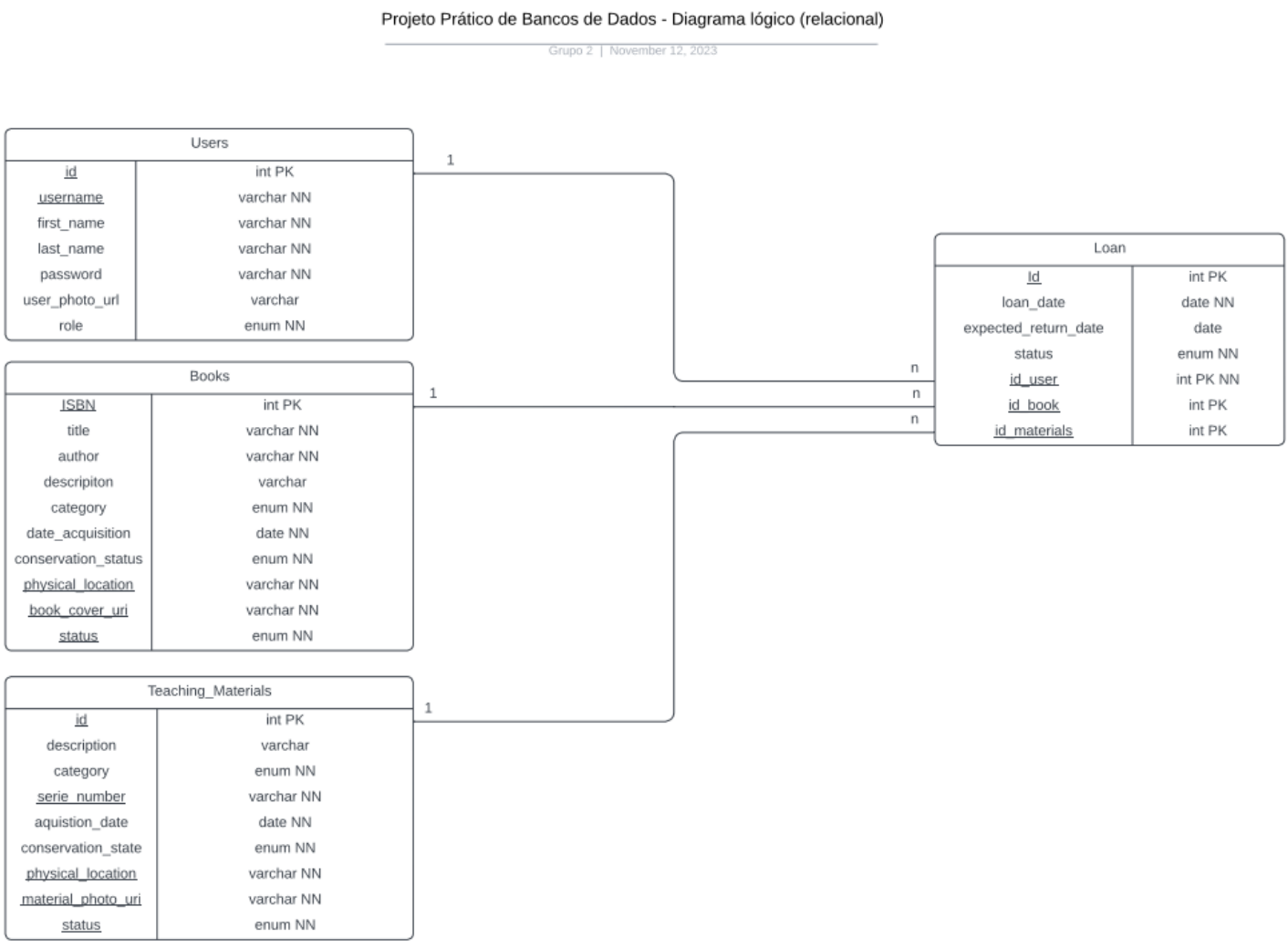
A tabela Books armazena os dados dos livros e também o status para indicar se ele está emprestado ou não, isso existe para impedir que ele seja emprestado ao mesmo tempo para 2 usuários diferentes.

A tabela Teaching_Materials tem uma função semelhante a tabela Books, essa também armazena os dados dos materiais e o status indicando se ele está emprestado ou não.

Por fim, temos a tabela Loan que registra os empréstimos realizados, as datas de empréstimo e devolução, os membros podem pegar apenas 1 livro ou 1 material por vez e só pode ser emprestado outro item quando a data de devolução for registrada. Ela também tem um trigger para checar a data de entrega e se o livro/material está disponível

Sempre que um empréstimo é realizado, o livro ou o material tem o status alterado imediatamente por trigger e quando o empréstimo é finalizado o trigger atualiza o status novamente.

Abaixo, temos o Diagrama lógico do banco, demonstrando o formato dos dados e as ligações entre as tabelas.



Descrição da API:

A api foi feita seguindo o padrão Active Record, para cada tabela temos uma classe com as funções de CRUD básicas e algumas diferentes para facilitar o uso. Todas as classes possuem na função __init__ uma definição do objeto com os mesmos atributos da tabela SQL correspondente, elas também possuem uma função interna _bd connect para criar a conexão com o banco.

Os métodos específicos de cada classe são esses aqui:

User:

- **authenticate_user**: verifica se o username e o password passado nos parâmetros da classe e retorna o usuário se existir.
- **create_user**: cria um novo usuário com os dados passado nos parâmetros da classe e retorna True/False.
- **update_user**: atualiza o usuário com os dados passado nos parâmetros da classe e retorna True/False.
- **delete_user**: excluir o usuário de acordo com o id passado nos parâmetros da classe e retorna True/False.
- **get_users**: retorna uma lista com todos os usuários do banco
- **get_user_by_id:** retorna o usuário com o id corresponde passado nos parâmetros da classe
- **get_all_members**: retorna todos os usuários com a permissão “MEMBER”

Book:

- **create_book**: cria um novo livro com os dados passado nos parâmetros da classe e retorna True/False.
- **update_book**: atualiza o livro com os dados passado nos parâmetros da classe e retorna True/False.
- **delete_book**: excluir o livro de acordo com o isbn passado nos parâmetros da classe e retorna True/False.
- **get_books**: retorna uma lista com todos os livros do banco
- **get_book_by_isbn:** retorna o livros com o isbn corresponde passado nos parâmetros da classe
- **get_available_books**: retorna todos os livros com o status “AVAILABLE”

TeachingMaterial:

- **create_teaching_material**: cria um novo material com os dados passado nos parâmetros da classe e retorna True/False.
- **update_teaching_material**: atualiza o material com os dados passado nos parâmetros da classe e retorna True/False.
- **delete_teaching_material**: excluir o material de acordo com o id passado nos parâmetros da classe e retorna True/False.

- **get_teaching_materials**: retorna uma lista com todos os materiais do banco
- **get_teaching_material_by_id**: retorna o material com o id corresponde passado nos parâmetros da classe
- **get_available_teaching_materials**: retorna todos os material com o status “AVAILABLE”

Loan:

- **create_loan**: cria um novo empréstimo com os dados passado nos parâmetros da classe e retorna True/False.
- **update_loan_status**: atualiza o status do empréstimo com os dados passado nos parâmetros da classe e retorna True/False. Se o novo status for “COMPLETED” a data de devolução será igual a hoje.
- **update_return_date:** atualiza a data de devolução do empréstimo com os dados passado nos parâmetros da classe e retorna True/False.
- **delete_loan**: excluir o empréstimo de acordo com o id passado nos parâmetros da classe e retorna True/False.
- **get_all_loans**: retorna uma lista com todos os empréstimos do banco
- **get_loans_by_id**: retorna o empréstimo com o id corresponde passado nos parâmetros da classe
- **get_in_progress_loans**: retorna todos os material com o status “IN_PROGESS”

Funcionalidades:

Os Usuários são divididos em 3 grupos, administradores e membros e os não logados.

O login de todos os usuários é feito encriptando a senha de input e comparando com a senha encriptada que vem do banco, se o username e a senha baterem o usuário será logado.

Os administradores podem acessar todas as funções da plataforma, sendo permitido realizar um CRUD em todas as tabelas do banco. Eles podem registar novos materiais e livros, cadastrar novos usuários, editar e excluir eles e também pode gerar empréstimos para membros.

Os membros podem apenas fazer o get no seu usuário, nos livros e materiais e também pode criar empréstimos para si mesmo.

Os usuários não logados podem apenas fazer o visualizar os livros e materiais (apenas get), as demais funções requerem login.

Quando um material ou livro for emprestado ele fica bloqueado para empréstimos até o anterior ser concluído.

Os empréstimos só podem ser atribuídos a um usuário membro mas podem ser criados por um administrador ou pelo próprio membro. Cada membro só pode ter 1 empréstimo ativo por vez, seja de livro ou material, quando o status do empréstimo anterior for “COMPLETED” ele pode realizar outro. O membro também pode adiar a data de devolução prevista.

Github do projeto: https://github.com/victorandre957/LabLibrary

Correções:

Realizada na entrega 3:

- Atualizamos a configuração do ambiente para incluir a utilização do Docker para rodar o banco de dados do projeto que havíamos esquecido.

- Adicionamos o link do github do projeto que estava faltando.

- Fizemos pequenas mudanças nos diagramas do banco de dados, adicionando atributos que não estavam incluídos inicialmente e uma pequena descrição do banco.

Realizadas na entrega 4:

- Adicionamos um trigger para mudar o status do livro ou do material quando um empréstimo for feito, para impedir que outra pessoa pegue o livro enquanto ele já está ocupado.
- Completamos o trigger que atualiza o status quando um empréstimo for realizado para mudar o status do material também.
- O trigger do empréstimo foi refatorado para não verificar a data do empréstimo, isso acontecerá no check dentro da tabela para garantir que a verificação ocorra no update também.
- O trigger para encriptar a senha foi atualizado para mudar a codificação
- Os Triggers das tabelas foram documentados.
- Atualizamos as configurações de ambiente, numpy e pandas foram removidos porque não houve necessidade.
- Adicionamos a descrição dos triggers do banco de dados.
- Adicionada licença MIT no repositório