FCTUC FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

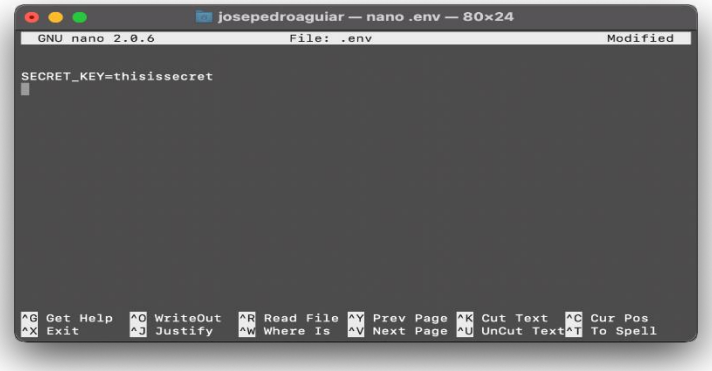
Relatório Projeto de Base de Dados

Trabalho realizado por:

Diogo Sebastião Cleto - 2019198370
João Miguel Ferreira Castelo Branco Catré - 2019218953
José Pedro Silva Aguiar – 201922462

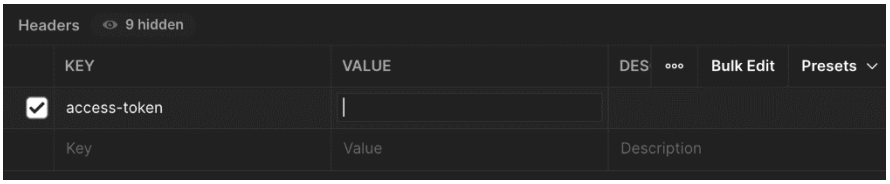
Manual de utilizador

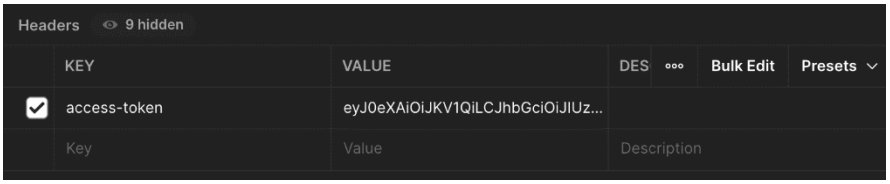
A base de dados utiliza uma variável de ambiente (*environment variable*) e, para tal, é necessário declará-la:



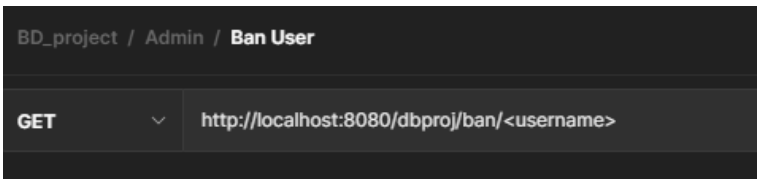
Para a maior parte dos *requests* desenvolvidos é necessário um *token*. Para isso, inicialmente, é necessário criar um (ou mais) utilizador, para que, posteriormente, seja possível efetuar o *login*.

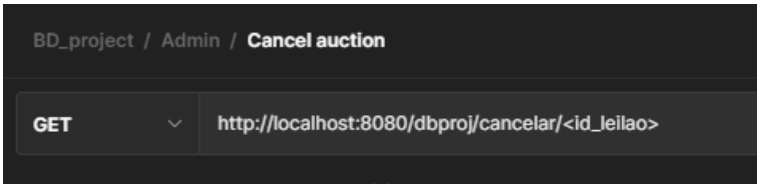
No *request* do *login* é obtido o *token* que será necessário colocar no *header* para que se possam realizar as funções.

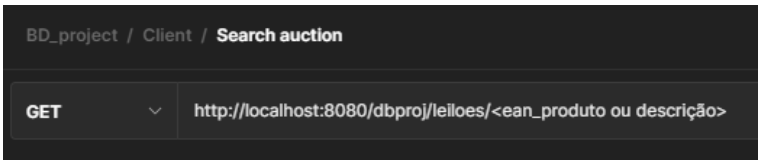


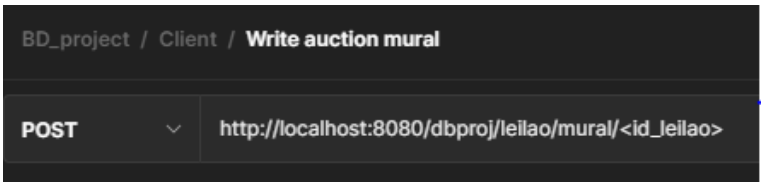


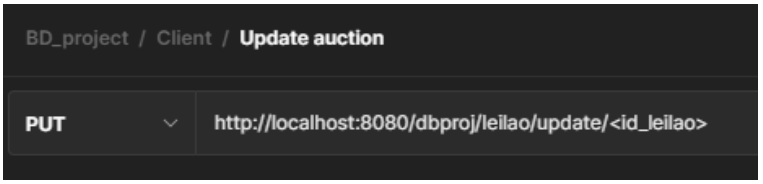
As seguintes funções: banir utilizador; cancelar um leilão; procurar leilão; escrever no mural do leilão; atualizar leilão; nova licitação e procura de leilão pelo id, contêm argumentos que têm de ser introduzidos. De seguida, exemplificamos como, com recurso às seguintes imagens, de acordo com a ordem dada acima:

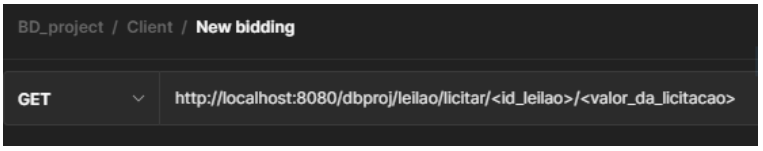


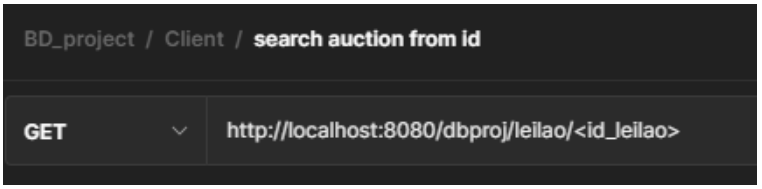












Manual de instalação

Primeiramente, é necessário a aplicação Docker. Esta pode ser obtida através do seguinte link:
https://www.docker.com/products/docker-desktop.

A linguagem utilizada para a *api* foi *Pyhton* e para a base de dados foi *postgreSQL*.

Para conseguir correr o programa há uma série de bibliotecas (da linguagem Python) necessárias para o bom funcionamento da aplicação. Sendo estas :

Flask, *jsonify* e *request*, que pertencem à biblioteca *flask*; *logging*; *time*; *psycopg2*; *decrypted*; *pytz*; *jwt*; *wraps* da biblioteca *functools*; *datetime* e *timedelta*, pertencentes à biblioteca *datetime*; *os* e, por último, *load_dotenv*, pertencente à biblioteca *dotenv*.
As mesmas já estão inseridas no *Docker file* do container da *api*.

Para instalar a base de dados e a api, apenas é necessário correr os containers. Para isso, utilza-se o seguinte comando: *docker-compose -f docker-compose-python-psql.yml up --build*.

Há, também, a necessidade de instalar a aplicação “Postman” ou de aceder à mesma na web, pelo link: https://www.postman.com/, sendo necessário importar as funções do ficheiro que se encontra na diretoria *postman*.

Diagrama ER

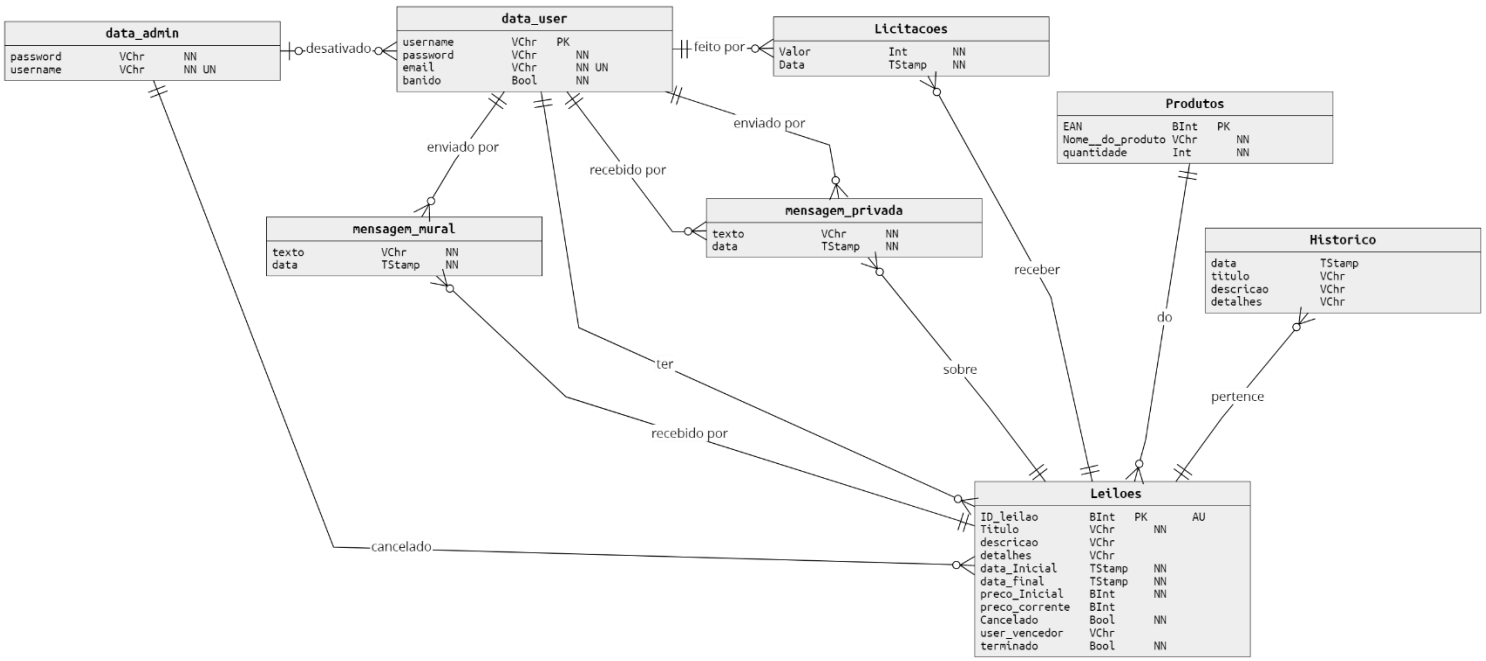
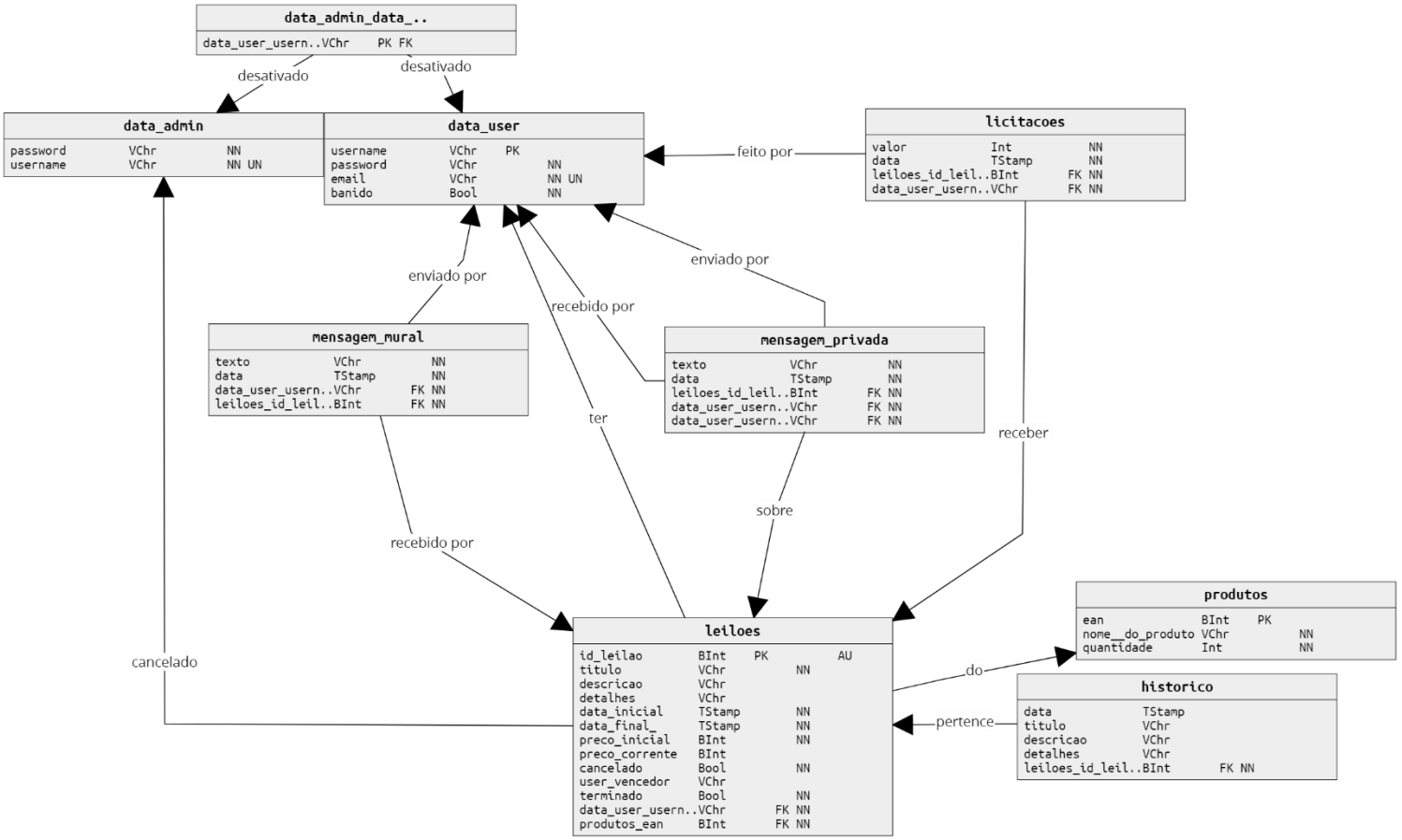


Diagrama Físico



Plano de trabalho

Dividimos os pontos do trabalho da seguinte maneira:

1. Criar um novo utilizador, inserindo os dados requeridos pelo modelo de dados.
2. Login com *username* e password, recebendo um *token* de autenticação em caso de sucesso, *token* esse que deve ser incluído nas chamadas subsequentes.
3. Criar um novo leilão.
4. Listar os leilões que estão a decorrer, obtendo uma lista de identificadores e descrições.
5. Pesquisar leilões existentes.
6. Consultar detalhes de um leilão.
7. Listar todos os leilões em que o utilizador tenha atividade.
8. Efetuar uma licitação num leilão.
9. Editar propriedades de um leilão.
10. Escrita de mensagens no mural de um leilão.
11. Entrega imediata de mensagens a utilizadores.
12. Notificação de licitação ultrapassada.
13. Término do leilão na data, hora e minuto marcados
14. O administrador cancelar um leilão.
15. O administrador banir permanentemente um utilizador.
16. O administrador obter estatísticas de atividade na aplicação

- Diogo Cleto:
  Pontos – 8, 10, 11, 14, 15 + relatório;
  Horas de trabalho – 50;
- João Catré:
  Pontos – 1, 3, 4, 6, 9 + relatório;
  Horas de trabalho – 50;
- José Aguiar:
  Pontos – 2, 5, 7, 12, 13, 16 + relatório;
  Horas de trabalho – 50.