MINI-PROJETO DE BASE DE DADOS | 2020-2021

Relatório de Projeto

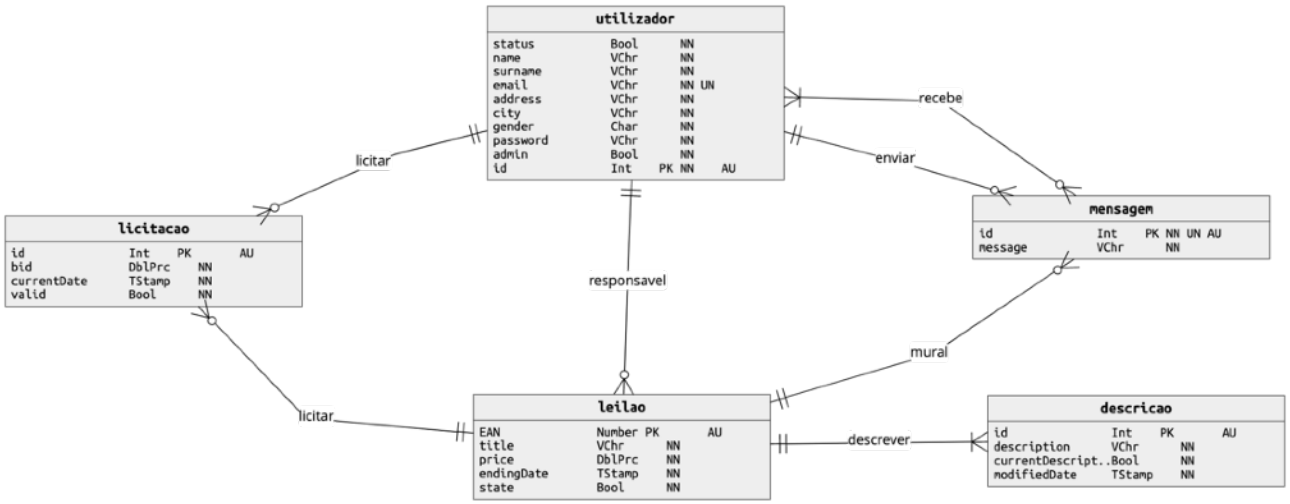
Davide Figueiredo Areias | 2019219422
Rui Eduardo Carvalho Marques | 2019216539
Thomas Pereira Fresco | 2019219057

1 2 9 0
UNIVERSIDADE D COIMBRA

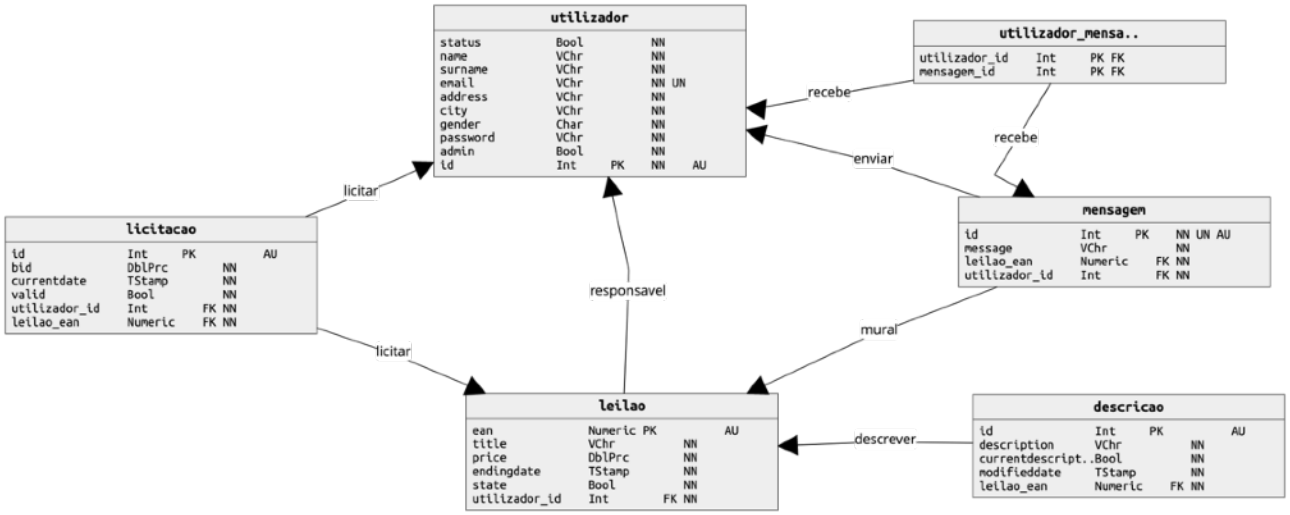
Licenciatura em Engenharia Informática

Diagramas finais

Conceptual



Physical



MINI-PROJETO DE BASE DE DADOS | 2020-2021

Plano de Desenvolvimento

Davide Figueiredo Areias | 2019219422
Rui Eduardo Carvalho Marques | 2019216539
Thomas Pereira Fresco | 2019219057

Licenciatura em Engenharia Informática

1 2 9 0
UNIVERSIDADE D COIMBRA

Plano de Desenvolvimento

A maioria do trabalho realizado foi sempre na presença dos 3 elementos (no mesmo local ou virtualmente) e, por essa razão, a grande maioria dos problemas que foram surgindo resolveram-se em conjunto e o tempo dedicado ao projeto foi o mesmo para todos os membros. Aproximadamente, cada um dos elementos despendeu de 60 horas para a realização do mini-projeto.

No entanto, é possível dividir as tarefas realizadas por cada membro da seguinte maneira:

- **Davide Areias:** implementação da *API* através do *Python* e a construção da maioria dos *requests* no "*Postman*".

- **Rui Marques:** desenvolvimento de *triggers* e funções associadas; construção da maioria das operações de *SQL*; indexação da base de dados.

- **Thomas Fresco:** essencialmente trabalho de reportagem ao longo do trabalho desenvolvido e constante teste, revisão e correção do código fonte e da *workspace* do "*Postman*". Desenvolvimento dos documentos "plano de desenvolvimento", manuais de instalação e de utilização.

Tanto o código como os *requests/endpoints* do *Postman* encontram-se organizados por secções. Essas secções dizem respeito a "Utilizador", "Administrador", "Leilões" e "Mensagens". Ainda referente ao código fonte em *Python*, é possível dividi-lo em 2 grandes grupos de funções: as relativas aos *endpoints* para o tratamento dos *requests* e as suas auxiliares. Esta solução de organização agilizou imenso a procura de porções de código.

Ao nível da segurança, as *passwords, quando facultadas para registo de um novo utilizador,* são encriptadas antes de serem armazenadas na base de dados, através do uso de funções da biblioteca do *python* "*werkzeug.security*". Quando um utilizador procede a um *login*, a palavra-passe inserida é encrostada com o mesmo método referido anteriormente e comparada com a palavra-passe armazenada que corresponde ao *email* do utilizador que tentou fazer o *login*. No ficheiro de código fonte, não existe qualquer função de desencriptação, o que contribui para uma maior segurança dos dados guardados.

Os últimos tópicos trabalhados foram a indexação e o tratamento de concorrência. Nesta fase, foram criados "*indexes*" para melhorar o desempenho temporal das "*queries*". Tentou-se contornar possíveis problemas de concorrência, mas não se conseguiu chegar a uma solução que não obtivesse *warnings* durante a execução.

O plano de desenvolvimento foi, na nossa opinião, bem sucedido, ainda que com lugar a melhoria, e, provavelmente, seria ainda mais eficaz numa situação em que apresentássemos mais experiência, tanto ao nível de programação com bases de dados, como em divisão das tarefas.

MINI-PROJETO DE BASE DE DADOS | 2020-2021

Manual de Instalação

Davide Figueiredo Areias | 2019219422
Rui Eduardo Carvalho Marques | 2019216539
Thomas Pereira Fresco | 2019219057

Licenciatura em Engenharia Informática

1 2 9 0
UNIVERSIDADE D COIMBRA

Manual de instalação

Para que o trabalho desenvolvido possa ser testado em qualquer computador, é necessário proceder ao *download*/instalação de diversos recursos.

- No material submetido pelo "Inforestudante", encontra-se uma pasta com o nome **"Projeto_BD"**, que contém todos os ficheiros necessários para o teste do trabalho;

- *Software* ***"Docker"***, que irá construir e colocar em *host* a base de dados e a *API* (em localhost);

- Se o sistema operativo for "*Windows*", deverá transferir e instalar o *software* ***"git for windows"***, que irá permitir emular uma consola "*BASH*" e correr o ficheiro ***"docker-compose-python-psql.sh"***, que irá dar ordens ao programa *Docker* para que crie a base de dados e a coloque em *host*.

- Uma vez que os testes foram todos realizados através do ***"Postman"***, aconselha-se fortemente o recurso a essa mesma ferramenta, embora não seja exatamente obrigatório. Para esse efeito, é possível aceder às "*collections*" (lista de *requests* associados a *endpoints* específicos) do ficheiro exportado a partir da aplicação ou diretamente do *browser*, que se encontra na pasta indicada no primeiro ponto (***"postman/Projeto.postman_collection.json"***). É possível, em alternativa, o teste do trabalho realizado ser feito com recurso ao ***"Curl"***.

MINI-PROJETO DE BASE DE DADOS | 2020-2021

Manual de Utilização

Davide Figueiredo Areias | 2019219422
Rui Eduardo Carvalho Marques | 2019216539
Thomas Pereira Fresco | 2019219057

Licenciatura em Engenharia Informática

1 2 9 0
UNIVERSIDADE D COIMBRA

Manual de utilização

O primeiro passo, após abrir o programa "*Docker*", deve ser executar o ficheiro "*docker-compose-python-psql.sh*". Este processo irá fazer com que a base de dados e a API sejam construídas e colocadas em *host*.

No programa "*Postman*", deve-se primeiramente importar o ficheiro "*Postman/Projeto.postman_collection.json*" para uma nova *workspace*. Na secção "*Environments*", clicar em "+" para criar um novo ambiente, dando-lhe um nome arbitrário. Nele, deve-se criar uma variável com o nome "*jwt_token*" e clicar em "*save*".

Depois de selecionar, no canto superior direito, o novo ambiente que foi criado, a *API* encontrar-se-á pronta a ser usada. Todos os *requests*, à exceção do registo de utilizadores e *print* dos mesmo (esta última não foi diretamente pedida no enunciado), apresentam verificação de *token* de *login*. A nova *token* obtida aquando o *login é* escrita por cima da anterior na sessão considerada (como variável de *environment*), fazendo com que o utilizador que anteriormente tinha feito *login* simule um *logout e* dê lugar ao novo utilizador para usufruir da nova sessão iniciada. Se o utilizador não tiver uma *token* atribuída e tentar executar algum *request*, recebe a mensagem "*token* is missing!". Uma token tem uma duração definida para 30 minutos, podendo ser alterada.

Para melhor perceção do que cada *request* faz e que proteções foram feitas para cada situação, encontra-se, abaixo, uma lista detalhada sobre as informações relevantes sobre cada um.

Lista de *endpoints*: (os *endpoints* que não foram diretamente pedidos no enunciado encontram-se assinalados com *)

Utilizador:

- **Registo de utilizadores.**

  Req: [POST] http://localhost:8080/dbproj/user/

  (address, city, email, gender, name, password, surname, admin)

  Res:

  - **"New user inserted!" -** em caso de insert com sucesso de um utilizador
  - **"Email already in use!" -** tentar dar insert de um utilizador com um email já existente na base de dados.
  - **"Fail to insert new user!" -** em caso de erro
  - **"New admin inserted!" -** em caso de insert com sucesso de um administrador

- **Autenticação de utilizadores.**

  Req: [PUT] http://localhost:8080/dbproj/user/

  (email, password)

  Res:

  - **"Login failed" -** se o utilizador introduzir uma palavra-passe incorreta.
  - **"User not registered!" -** se o utilizador não estiver registado.
  - **"User already logged in!" -** se o utilizador já estiver logged in.
  - **{"Login with Success!" : token} -** em caso de sucesso.
  - **"Failed!" -** em caso de erro.

- **Listar todos os leilões em que o utilizador tenha atividade.**

  Req: [GET] http://localhost:8080/dbproj/user/activity/

  Res:

  - **"No activity yet!" -** se o utilizador não tiver atividade.
  - **Lista dos leilões em que tem atividade -** se o utilizador tiver atividade em pelo menos um leilão.

- **Efetuar uma licitação num leilão.**

  Req: [POST] http://localhost:8080/dbproj/licitar/<leilaoId>/<licitacao>

  Res:

  - **"Error: Admins cant bid!" -** utilizador é administrador e tenta licitar num leilões
  - **"Error: You own this auction!" -** utilizador tentar licitar num dos seus leilões
  - **"Error: Auction does not exist!" -** se <leilaoId> não corresponder a nenhum leilão
  - **"Error: Auction ended!" -** utilizador tenta licitar num leilão que já acabou
  - **"Bid Inserted!" -** em caso de sucesso
  - **"Bid too low!" -** se o valor da licitação for igual ou inferior ao atual maior
  - **"Fail to insert bid!" -** em caso de erro

Leilão:

- **Listar todos os leilões existentes.**

  Req: [GET] http://localhost:8080/dbproj/leiloes/

  Res:

  - **Lista vazia -** se não houver leilões ativos
  - **Lista dos leilões ativos (ean e description) -** se houver pelo menos um leilão ativo

- **Pesquisar leilões existentes.**

  Req: [GET] http://localhost:8080/dbproj/leiloes/<keyword>

  Res:

  - **“Auction not found!” -** se <keyword> não corresponder a nenhum leilão (ean ou description)
  - **Lista dos leilões com match da <keyword> (ean ou description) -** se pelo menos um leilão verificar essa condição.

- **Consultar detalhes de um leilão.**

  Req: [GET] http://localhost:8080/dbproj/leilao/<leilaoId>

  Res:

  - **Lista vazia -** se <leilaoId> não corresponder a nenhum leilão
  - **Lista de ean, title, price, endingdate, state, utilizador_id, description -** se <leilaoId> corresponder a um leilão

- **Criar um novo leilão.**

  Req: [POST] http://localhost:8080/dbproj/leilao/

  (ean, title, price, endingdate, description)

  Res:

  - **“Admins cant create auctions!” -** utilizador é administrador e tenta criar um leilão
  - **"Inserted!" -** em caso de insert com sucesso de um novo leilão
  - **“Failed!” -** em caso de erro (ex: ean repetido)

- **Editar propriedades de um leilão.**

  Req: [POST] http://localhost:8080/dbproj/leilao/<leilaoId>

  (description)

  Res:

  - **“Auction not found!” -** se <leilaoId> não corresponder a nenhum leilão
  - **"You don't own this auction!” -** o utilizador não pode alterar a descrição por não ser dono
  - **'Auction description edited with success’ -** em caso de sucesso
  - **'Failed to edit Auction!’ -** em caso de erro

- **Término do leilão na data, hora e minuto marcados.**

  Req: [PUT] http://localhost:8080/dbproj/leilao/time_end/

  Res:

  - **“Ended: %d Auction(s).” -** se houver pelo menos um leilão com instante de término inferior ao instante atual.
  - **“No auctions to expire!” -** se não houver leilões com instante de término inferior ao estante atual.

Mensagem:

- **Escrever mensagem no mural de um leilão.**

  Req: [POST] http://localhost:8080/dbproj/message/<leilaoId>

  (message)

  Res:

  - **“Message Sent!” -** em caso de sucesso
  - **“Failed to send message” -** em caso de erro (ex: <leilaoId> não corresponde a nenhum leilão)

- **Mostrar caixa de correio de um utilizador.***

  Req: [GET] http://localhost:8080/dbproj/mailbox/

  Res:

  - **"Mailbox Empty!” -** se o utilizador não tiver mensagens na caixa de correio.
  - **Lista das mensagens (ean, id_remetente e message) -** se o utilizador tiver pelo menos uma mensagem na caixa de correio.

- **Mostrar mural de um leilão.***

  Req: [GET] http://localhost:8080/dbproj/feed/<ean>

  Res:

  - **“Auction does not exist!” -** se <leilaoId> não corresponder a nenhum leilão.
  - **Lista das mensagens (utilizador_id e message) -** se o leilão tiver pelo menos uma mensagem no mural.

Admin:

- **Um administrador pode obter estatísticas de atividade na aplicação.**

  Req: [GET] http://localhost:8080/dbproj/stats/

  Res:

  - **“You are not Admin!” -** o utilizador não é administrador
  - **Top 10 utilizadores com mais leilões criados [id utilizador, nº de leilões criados] (podendo ser vazio),**
    top 10 utilizadores que mais leilões venceram [id utilizador, nº de vitórias] (podendo ser vazio),
    número total de leilões nos últimos 10 dias (podendo ser 0).

- **Um administrador pode cancelar um leilão.**

  Req: [PUT] http://localhost:8080/dbproj/leilao/force_end/

  (ean)

  Res:

  - **“You are not Admin!” -** o utilizador não é administrador
  - **“Auction does not exist!” -** se o leilão não existir.
  - **“Failed to end auction!” -** em caso de erro, se o leilão ja tiver sido encerrado ou se não existir.
  - **“Forced auction desactivation!” -** em caso de sucesso.

- **Um administrador pode banir permanentemente um utilizador.**

  Req: [PUT] http://localhost:8080/dbproj/ban/<user_id>

  Res:

  - **“You are not Admin!” -** o utilizador não é administrador
  - **“User %d banned!” -** o utilizador <user_id> é banido com sucesso.
  - **“Fail to ban user!” -** o utilizador não é administrador

- **Mostra todos os utilizadores registados.***

  Req: [GET] http://localhost:8080/dbproj/user/

  Res:

  - **Lista dos utilizadores -** se houver pelo menos um utilizador
  - **Lista vazia -** se não houver utilizadores