UNIVERSIDADE DE SÃO PAULO - USP
BACHARELADO EM SISTEMAS DE INFORMAÇÃO
SSC0541 - LABORATÓRIO DE BASE DE DADOS

PROJETO FINAL

Documentação externa

GRUPO 9:
JÚLIO CÉSAR CABRAL - 13672922
LUCAS MASAKI MAEDA- 13692272
MATHEUS HENRIQUE DA SILVA - 13696658
OTÁVIO AUGUSTO COLUCCI DE OLIVEIRA - 13692421

São Carlos - SP
09/06/24

DOCUMENTAÇÃO EXTERNA:

Ferramentas utilizadas (e versões):

- Pycharm Version Community 2024.1.3
- SQL Developer
- Visual Studio Code

Linguagens utilizadas (e versões):

- Python 3.10.

APIs utilizadas (e versões):

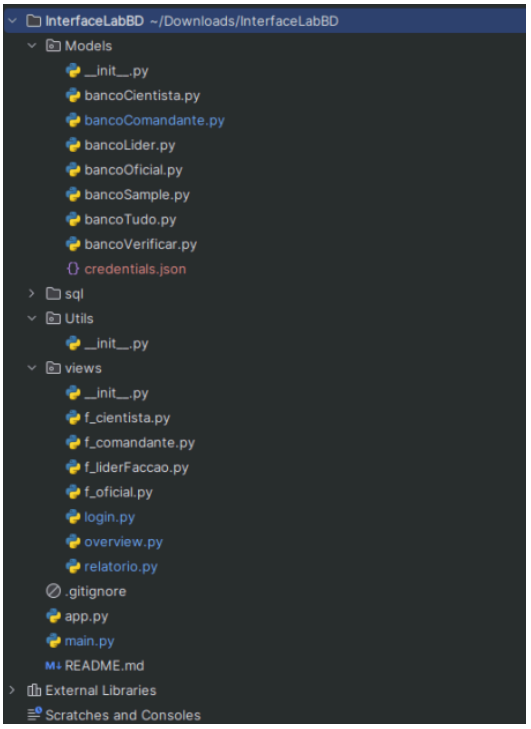
- Drive da OracleDB para Python 2.2.1;

Descrição dos arquivos SQL:

- **esquema.sql:** Arquivo com o script para a criação de todas as tabelas
- **features.sql:** Arquivo com algumas features implementadas na aplicação, tais como os índices para a melhora da complexidade nas buscas.
- **funcionalidades.sql:** Arquivo com as procedures de funcionalidades de todos os tipos de usuários, separado devidamente. Todas as funcionalidades estão dentro de um package.
- **inserts.sql:** Arquivo com inserções utilizadas para testar a aplicação como um todo.
- **relatorios.sql:** Arquivo com views dos relatórios e procedures para chamar as consultas para dentro da interface.
- **testes.sql:** Arquivo com usuários interessantes para testes (combinações de cargos e se é líder de facção ou não) além de todos os casos de teste para cada funcionalidade. Colocamos para cada funcionalidade, um caso de sucesso e um caso de erro para cada exceção, e os dados utilizados para cada caso.
- **users.sql:** Arquivo com diversas features voltadas para a parte de usuário/login/overview. Existe uma seção com a criação das tabelas e views utilizadas (aqui criamos uma tabela de usuários, uma tabela de log, e uma view com as informações importantes do usuário), uma seção com a criação dos triggers (um para a encriptografia das senhas com MD5, como solicitado, e outro para “sincronizar” a tabela de líderes com a de usuários) e uma seção com procedures (uma para criar usuário, outra para colocar os lideres que não são usuários como usuários, uma para passar os dados de login para a interface, uma para passar os dados de overview para a interface, e uma para inserir os logs na tabela).

Descrição dos demais arquivos, voltados para a interface:

- **Models:** onde estão as funcionalidades
- **Views:** onde estão as telas e partes gráficas
- **App:** onde fica a aplicação em si
- **Main:** apenas testes da aplicação



- **readme.md:** Estamos enviando também um readme com as instruções para executar a aplicação.

Requisitos obrigatórios:

- **View com junção (justificativa):** Utilizamos várias views na aplicação, uma utilizada para o login/overview e as demais utilizadas para os relatórios. Cada view teve seu uso totalmente justificado:
    - A view de login/overview, serve para puxar todos os dados necessários para o login na plataforma quanto para o overview, guiando as condicionais de ações dentro da aplicação. Isso é, um usuário pode ser ou não um líder de facção, além de ter um cargo (comandante, cientista ou oficial), porém para conferir isso tudo precisamos fazer várias junções, sendo assim, colocamos tudo na view.
    - Para as views de relatórios, o mesmo acontece em uma escala maior. Isso é, a maioria dos relatórios exige a junção da maioria das tabelas, o que pode deixar a complexidade da busca pesada, logo buscamos em uma view sempre que geramos os relatórios.
- **Índices:** Utilizamos índices para a melhoria da complexidade de algumas buscas, principalmente as encontradas em relatórios, porque, como falado anteriormente, exigem a junção de várias tabelas.

- **Procedimentos/funções com pacotes:** Basicamente todas as funcionalidades do nosso projeto são procedimentos e funções. As funcionalidades em si servem para alterar o banco como um todo, seja inserir, remover, atualizar uma tupla, entre outros. Porém utilizamos várias procedures também para passar dados para a interface, visto que as operações devem ser realizadas somente via interface. Para essas, utilizamos um cursor, salvando a consulta dentro do cursor aberto, onde a interface pega os dados e fecha o cursor.

- **Triggers:** Utilizamos alguns triggers no projeto, alguns para tratamento de erros e alguns voltados para usuários. Dois que foram bastante utilizados: um para a encriptografia das senhas com MD5, como solicitado, e outro para "sincronizar" a tabela de líderes com a de usuários

- **Tratamento de erros e exceções:** Como comentado no arquivo de testes, todas as funcionalidades e relatórios, criados com procedures e funções, tiveram as exceções tratadas, retornando erros para o usuário da forma mais clara e transparente o possível. Geramos casos de teste para cada funcionalidade. Colocamos para cada funcionalidade, um caso de sucesso e um caso de erro para cada exceção, e os dados utilizados para cada caso.

- **Transações:** Definimos uma transação dentro da procedure de inserir logs, para ter uma maior confiabilidade do registro de logs, recuperando os savepoints e dando rollback caso necessário. O teste dessa transação é feito dentro da procedure FUNCLIDER_ALTERARNOMEFACCAO por exemplo.

- **Operações realizadas somente via interface gráfica:** Como solicitado no enunciado, as operações são realizadas somente na interface. Isso é, toda e qualquer operação é feita dentro do banco e passada para a interface por meio de cursores.

Funcionamento da plataforma e telas:

Basicamente teremos 4 principais telas:

- **Tela de login:** Usuário insere ID e senha e entra na plataforma caso estiver certo;

- **Tela de overview:** Mostra os dados do usuário e dois botões: um para acessar as funcionalidades de sua função e um para acessar os relatórios de sua função. Clicando em cada um dos botões, ele redireciona o usuário para a interface respectiva.

- **Tela de funcionalidades:** Mostra todas as funcionalidades disponíveis para o usuário. Isso é, mostra as funcionalidades referentes ao cargo (se o usuário é cientista ele pode criar estrelas por exemplo) e mostra também as funcionalidades para um líder de facção, caso esse usuário seja um líder de facção.

- **Tela de relatórios:** Mostra todos os relatórios disponíveis para o usuário. Isso é, mostra os relatórios referentes ao cargo (se o usuário é cientista ele pode ver dados sobre estrelas por exemplo) e mostra também os relatórios de um líder de facção, caso esse usuário seja um líder de facção.