UNIVERSIDADE D
COIMBRA

Duarte Carvalho - 2020226658

Pedro Moreira - 2020230591

MeDElsync

Bases de Dados

Licenciatura em Engenharia Informática

2023/2024

Índice

Introdução

Este projeto é um projeto a ser realizado no âmbito da unidade curricular de Bases de Dados, da Licenciatura em Engenharia Informática, e tem como objetivo a criação de um sistema de gestão hospitalar de modo a pôr em prática os conhecimentos adquiridos ao longo deste semestre acerca desta matéria.

Este sistema deverá ser capaz de gerir tudo dentro do hospital, seja a atribuição de médicos a cirurgias, as doses de medicamentos administrados a uma pessoa internada, a faturação de cada internamento, etc, com as devidas funcionalidades descritas no enunciado e sem que haja incoerências de dados.

Membros do grupo

Duarte Rosa Carvalho – 77duarte77@gmail.com

Pedro José loureiro Moreira – pedromoreira0700@gmail.com

Potenciais conflitos de concorrência

Neste projeto, alguns dos potenciais conflitos que poderão haver serão:

-na marcação de consultas ou hospitalizações. Isto poderá acontecer quando e pessoas, ao atualizar o sistema, reservam médicos ou enfermeiros consultas com pacientes diferentes mas à mesma hora, por exemplo.

-nos dados de faturação. Se consultas ou hospitalizações forem adicionadas simultaneamente à ficha de uma pessoa, irá haver conflitos no valor a ser pago.

-na atualização de prescrições. Se esta ação não for controlada, poderá haver conflitos nos medicamentos prescritos ou nas quantidades/dosagens.

-na atualização da informação dos médicos e enfermeiros. Se atualizações forem feitas em simultâneo sem controlo, poderão haver conflitos na especialização ou hierarquia dos médicos ou enfermeiros, respetivamente.

Para além disso, é importante que, ao inserir dados distribuídos por várias entidades, seja garantido que, ou seja tudo inserido em todas as entidades envolvidas, ou não seja nada inserido em nenhuma.

Para controlar isso, será necessário ter utilizar transações e fazer os devidos bloqueios para controlar acessos a registos e tabelas de modo a não haver incoerência nos dados.

Distribuição de trabalho

Não foi feita propriamente uma divisão de tarefas. Ao longo que o trabalho foi sendo feito, foram-se fazendo os endpoints que faltavam sendo que, ao longo do trabvalho, algum elemento, a certa altura, atualizava tanto endpoints como o diagrama ER. Porém, o nome da pessoa principalmente responsável pela criação de um endpoint encontra-se à frente de cada um destes, no Manual de Utilização.

Manual de Instalação

1) Criação a base de dados no pgadmin

Abra o pgadmin e crie uma base de dados com o nome *MeDEIsync*.

Crie um utilizador com username=projeto_bd e password=password com permissões para aceder à base de dados então criada.

Dentro desta base de dados, aceder à query tool e correr o script SQL gerado pelo diagrama ER, *ER_FINAL_DO_FINAL.json*, que irá criar as entidades referentes à base de dados do MeDEIsync.

Após a criação da base de dados, aceder, novamente, à query tool e correr o script para a criação dos triggers utilizados na aplicação, armazenado no ficheiro *'CREATE TRIGGER appointment_bill BEF.txt'*.

2) Instalação das packages necessárias para correr o MeDEIsync

Abra o terminal e corra os seguintes comandos:

-pip install PyJWT

-pip install flask

-pip install psycopg2

3) Configuração do postman

Abra o postman e, na seccção *My workspace*, clique em *import* e import o ficheiro *Postman_collection_MeDEIsync*.

Após todos estes passos cumpridos, a aplicação está, então, pronta a ser utilizada.

Manual de Utilização

Uma vez cumpridos todos os procedimentos abordados no ponto anterior, para correr a aplicação basta executar o programa MeDEIsync.py. Após executar o programa, é necessário aceder ao postman e enviar o pedido denominado de *start*, que tem como objetivo popular a base de dados.

Estando a aplicação já a correr e a base de dados populada, pare efeitos de teste, esta suporta as seguintes funcionalidades:

Todos os pedidos serão enviados com a configuração {pedido_postman}→body→row.

-Adicionar Pacientes, Médicos, Enfermeiros e Assistentes (Duarte)

Para criar utilizadores, devemos então aceder ao postman e enviar um pedido POST correspondente ao utilizador que queremos adicionar à base de dados (Add patient, Add doctor, Add nurse, Add assistant), uma vez que cada tipo de utilizador terá diferentes atributos.

Os diferentes tipos de pedidos e atributos são:

-Para um paciente:

{"cc":"45678",

"nome":"paciente1",

"password":"paciente1password",

"data_nascimento ":"2003-01-11",

"medical_record ":"clean",

"n_utente":"123456",

"nib":"PT5000282998832",

"email":"pacient1@gmail.com"

}

-Para um médico:

{"cc":"12345",

"nome":"medico1",

"password":"medico1password",

"data_nascimento ":"1980-02-20",

"n_utente":"54321",

"nib":"PT981277893417",

"email":"medico1@gmail.com",

"contract ":"no rights",

"medical_license ":"licensed by UC!",

"main_specialization ":"dentist"

}

-Para um enfermeiro:

{"cc":"5678",

"nome":"enfermeiro1",

"password":"enfermeiro1password",

"data_nascimento ":"1970-03-15",

"n_utente":"6765",

"nib":"PT989827189739",

"email":"enfermeiro1@gmail.com",

"contract ":"nurse2win",

"internal_hierarchy ":"slave",

}

-Para um assistente:

{"cc":"34567",

"nome":"assistente1",

"password":"assistente1password",

"data_nascimento ":"1970-05-26",

"n_utente":"76543",

"nib":"PT509832098340",

"email":"assistente1@gmail.com",

"contract ":"my soul for a ferrari",

"field_0 ":"0"

}

Qualquer um destes pedidos irá retornar o *status_code*, erros (se ocorrerem) e o id do utilizador inserido (caso seja bem sucedido).

-Autenticacar utilizadores (Duarte)

Para autenticar um utilizador, envia-se um dos pedidos PUT existentes na coleção do postman, relativos a Login.

-exmplo de um pedido de autenticação:

{"username":"medico1",

"password":"medico1password"

}

Neste caso, a resposta ao pedido conterá o *status_code*, os erros (caso ocorram) e um token de autenticação gerado através do JWT (em caso de sucesso), que serverá para controlar as operações feitas por cada utilizador e o tempo de seção.

Posteriormente, em cada pedido ter-se-á que colocar o respetivo token obtido no campo *Token,* presente na secção *Authorization* de cada pedido no postman.

-Marcar appointments (Duarte)

Para marcar um appointment, é feito o pedido POST *Schedule Appointment* presente na coleção do postman. Este é uma funcionalidade apenas permitida a pacientes.

-exemplo do pedido:

{"doctor_cc":"1234",

"date":"2024-06-02",

"type":"ver as unhas"

}

Este pedido receberá como resposta o *status_code,* erros (caso existam) e o *id* desse appointment (caso seja bem sucedido)

-Ver appointments (Pedro)

Para listar os appointments e respetivos detalhes de um determinado paciente. Este endpoint apenas pode ser utilizado pelos assistentes ou pelo respetivo utilizador. Este será um pedido GET. Executar o pedido *See Appointments* do postman.

Este pedido obterá como resposta, para além do *status_code* e dos erros, caso ocorram, uma list com o *id* de cada appointment, o *id* do médico responsável por esse appointment e a *data* deste, em caso de sucesso.

-Marcar cirurgias (Pedro)

Para fazer uma marcação de cirurgia, basta escolehr um dos pedidos POST *Schedule surgery* presentes na coleção do postman. Neste caso, o atributo *hospitalization_id* pode, ou não, ser passdo. Caso seja, esta cirutgia será associada a uma hospitalização já existente, caso contrário uma nova hospitalização será criada. Para isso, é utilizado um trigger. Este é um pedido permitido apenas a assistentes.

Exemplos de pedidos de marcação de cirurgias são:

-caso o pedido não contenha um *id* relativo a uma hospitalização:

{"patient_id ":"4567",

"doctor":"1234",

"nurses":[["2345","responsible"],["5678","just learning"]],

"date":"2024-06-02",

"duration":"2",

"result":"open head"

}

-caso contenha *id* de uma hospitalização já existente:

Igual, mas adicionar o *id* no fim do endpoint.

A resposta a este pedido conterá, novamente, o *status_id*, erros (caso existam) e as informações relativas a essa cirurgia, tais como: *hospitalization_id, surgery_id, patient_id, doctor_id* and *date*.

-Ver prescrições (Duarte)

Para listar as prescrições relativas a um paciente, deve-se enviar o pedido *Get prescription* existente no postman, com o *person_id* do paciente em questão no final do endpoint. Apenas o próprio paciente ou os trabalhadores do hospital têm acesso a esta funcionalidade.

A resposta a este pedido conterá, para além do *status_code* e dos erros (caso ocorram), uma lista com as prescrições relativas ao paciente em questão e respetivas informações (id, *validity, dose* de cada medicamento, *frequência, medicine, etc*), em caso de sucesso.

-Adicionar Prescrições (Pedro)

Para adicionar prescrições ao sitema, deve ser utilizado o pedido POST *Add prescription* presente no postman. As prescrições poderão ser necessárias quando há um appointment ou uma hospitalização e estas só podem ser criadas por médicos.

Os pedidos deste tipo seguirão a seguinte estrutura:

-Exemplo de pedido para adicionar prescrições:

{"type ":"hospitalization",

"event_id":"7",

"validity":"2024-05-23",

"medicines":[{"medicine":"ben-u-ron", "dosage":"ig",

"frequency":"every 8 hours"}, {"medicine":"brufen",

"dosage":"0,5g", "frequency":"4 hours after ben-u-ron"}],

"patient_id":"4567"

}

A resposta a este pedido conterá, para além de, novamente, o *status_code* e os erros (caso ocorram), o *id* correspondente a essa prescrição.

-Executar pagamentos (Pedro)

Para executar pagamentos, ter-se-á de enviar o pedido POST *Pay bill* do postman. Esta funcionalidade está apenas disponível para pacientes e este pode ser feito em várias parcelas. Quando o pagamento fica completo, o status da *bill* é atualizado para *pago*.

-Exemplo de um pedido Pay bill

{"date ":"2024-05-23",

"ammount ":"2",

"method":"ATM"

}

A resposta a este pedido, para além do *status_code* e dos eventuais erros, conterá, também, a quantia que falta ser paga, em caso de sucesso.

-Listar top 3 (Pedro)

Este pedido serve para listar o top 3 de utilizadores na plataforma, tendo em conta o dinheiro gasto nesse mês.

Para utilizar esta funcionalidade, basta enviar o pedido GET *Top 3 patients* existente no postman. Este pedido apenas podes ser feito por assistentes.

Este pedido retornará, para além do *status_code* e dos erros, uma lista com as informações relativas aos pacientes que mais dinheiro gastaram nesse mês no hospital.

-Resumo diário (Pedro)

Este pedido serve para obter uma lista das hospitalizações, e respetivos detalhes, de um dado dia. Para realizar este pedido, basta enviar o pedido *GET Daily summary* presente no postman, que contém uma certa data (yyyy-mm-dd) e é apenas permitido a assistentes.

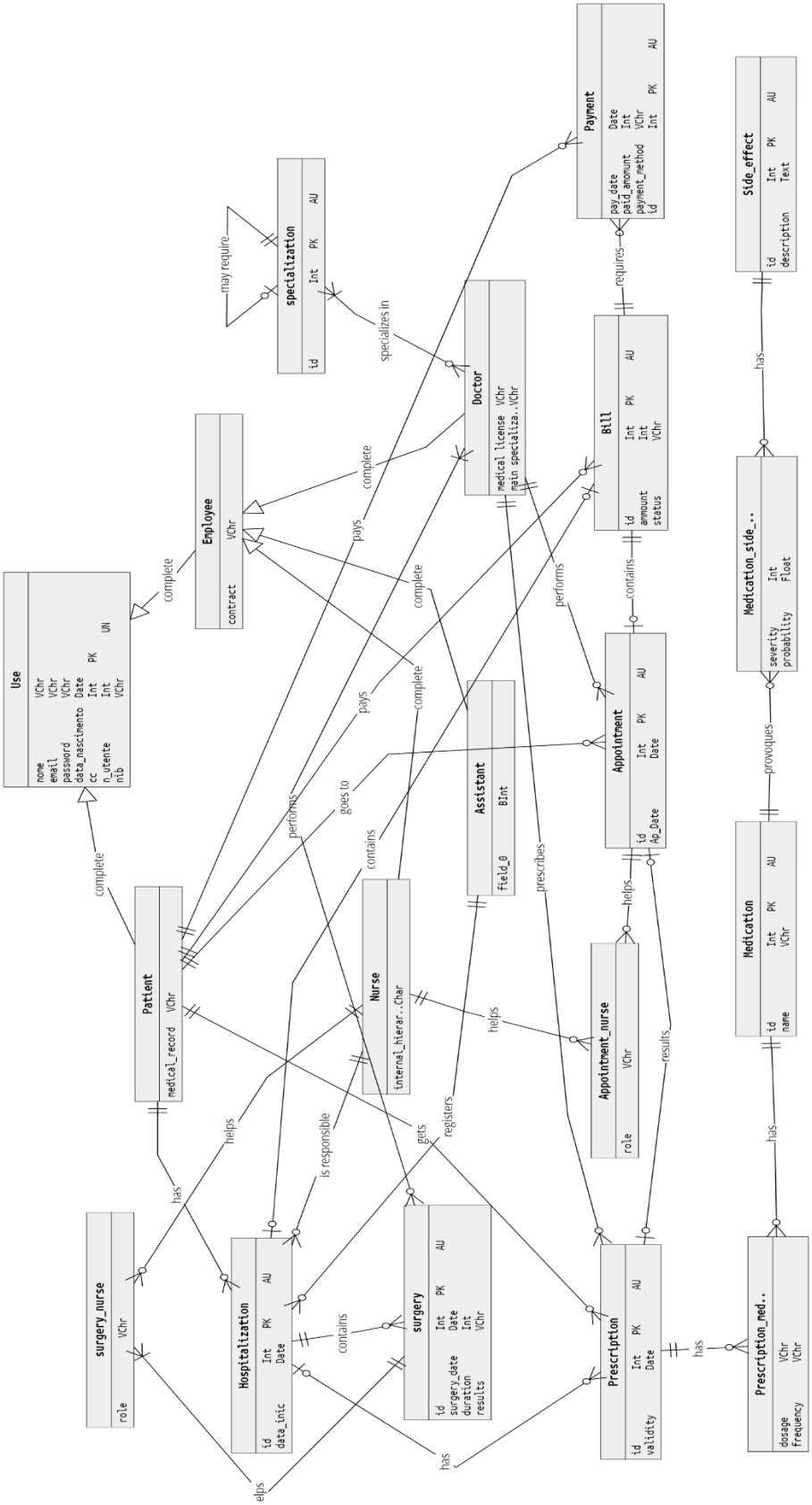
Este obterá uma resposta contento *status_code, eventuais* erros e uma lista com a quantidade gasta nesse dia, o numero de hospitalizações e prescrições.

-Gerar relatório mensal

Por fim, é também permitido, apenas a assistentes, gerar relatórios mensais, que serve para ir buscar uma lista com os medicos que mais cirurgias efetuaram em cada um dos últimos 12 meses. Para realizar esta funcionalidade, basta enviar o pedido *GET Monthly report* previamente criado no postman.

Este pedido recebe como resposta uma lista com 12 índices, em que cada um conterá o mês, o nome do médico com mais cirurgias nesse mês e o número de cirurgias efetuadas, para além do *status_code* e de eventuais erros.

Diagrama ER


Use
nome VChr
email VChr
password VChr
data_nascimento Date
cc Int PK
n_utente Int
nib VChr UN
Employee
contract VChr
Patient
medical_record VChr
Doctor
medical license VChr
main specializa..VChr
specialization
id Int PK AU
Payment
pay_date Date
paid_amount Int
payment_method VChr
id Int PK AU
Bill
id Int PK AU
amount Int
status VChr
Appointment
id Int PK AU
Ap_Date Date
Assistant
field_0 BInt
Nurse
internal_hierar..Char
Appointment_nurse
role VChr
surgery_nurse
role VChr
Hospitalization
id Int PK AU
data_inic Date
surgery
id Int PK AU
surgery_date Date
duration Int
results VChr
Prescription
id Int PK AU
validity Date
Prescription_med..
dosage VChr
frequency VChr
Medication
id Int PK AU
name VChr
Medication_side_..
severity Int
probability Float
Side_effect
id Int PK AU
description Text
may require
specializes in
requires
has
complete
pays
performs
contains
goes to
prescribes
helps
provoques
results
registers
gets
is responsible

Modelo físico