Hospital Management System

Relatório de Base de Dados

Mariana Sousa - 202221599
Rui Oliveira - 2022210616
Tiago Marques - 2022210638

Licenciatura em Engenharia Informática
Faculdade de Ciências e Tecnologia da Universidade de Coimbra

28 de maio de 2024

Conteúdo

1 Installation Manual

1.1 Introdução

Para este projeto, é necessário obter algumas ferramentas específicas. Para facilitar a compreensão do uso das mesmas, elaboramos este documento que irá proporcionar uma melhor compreensão.

1.2 Linguagens de Programação Necessárias

- Python
- SQL e pgSQL

1.3 Sistema de Gestão da Base de Dados

- PostgreSQL

1.4 Bibliotecas Utilizadas

- flask
- logging
- psycopg2
- datetime
- jwt
- hashlib

1.5 Outras Tecnologias

- Onda
- Postman

1.6 Instalação de Ferramentas

Antes de começar a elaborar o projeto, é necessário verificar se temos todas as bibliotecas instaladas. Caso contrário, será necessário instalá-las. De seguida, apresentamos um guia com os comandos para verificar e instalar.

- `pip install flask`
- `pip install psycopg2`

Ao executar estes comandos, se os detalhes de cada pacote aparecerem, significa que o pacote está instalado. Caso contrário, uma mensagem aparecerá que o pacote não está disponível.

Relativamente ao `SQL`, é necessário instalar, se possível, a última versão do `PostgreSQL` ou atualizá-lo.

Para a utilização do `Postman`, também é necessário instalá-lo, pode ser feito a partir de uma página *web*.

1.7 Configurações da Base de Dados

Para acessar a base de dados através do `psql` ou `pgadmin4`, é necessário configurar o acesso com *username* e *password* à escolha. A *porta* e o *localhost* já estão pré-definidos e não precisam ser modificados, a menos que necessário. Após essa configuração, é possível criar a base de dados para este projeto.

1.8 Mais Informação

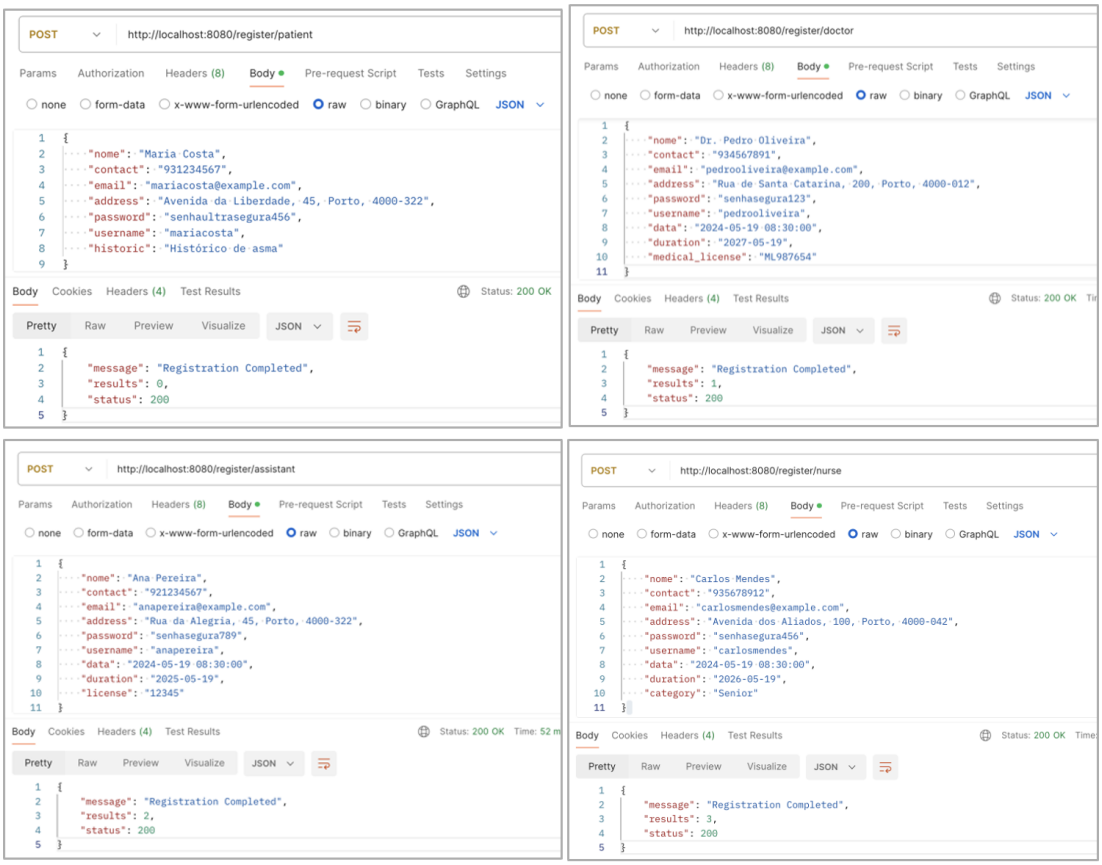
- https://www.python.org/
- https://www.postgresql.org/
- https://www.postman.com/

2 User Manual

2.1 Add Patient, Doctor, Nurse, and Assistant

Cria um novo utilizador(person), tendo em consideração o seu tipo (patient, assistant, nurse ou doctor), cada um apresenta diferentes atributos.

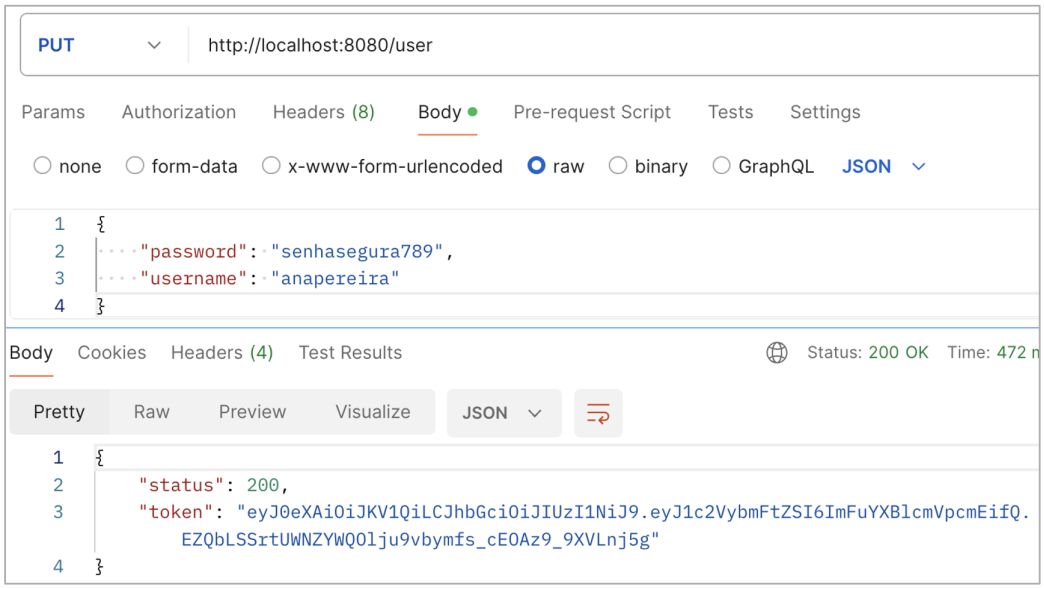
- URL: /register/<person_type>
- Método: POST



2.2 User Authentication

Para que cada utilizador realize o LOGIN, é necessário fornecer o seu "username"e respetiva "password", caso seja bem-sucedido é devolvido um 'token' que é necessário para as restantes funcionalidades.

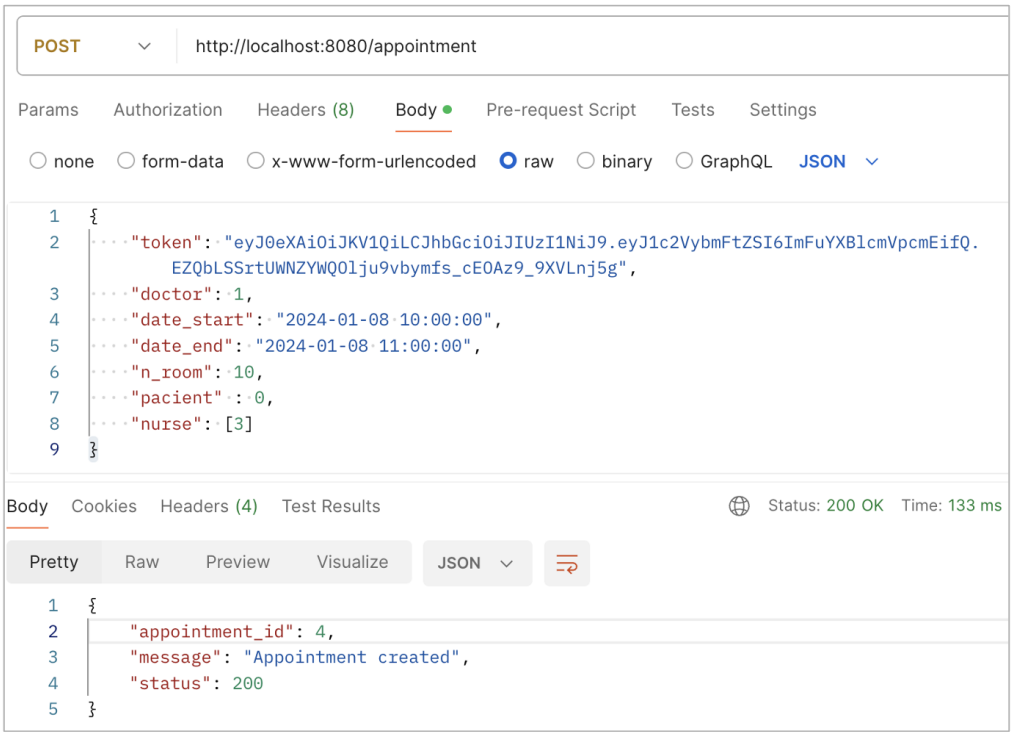
- URL: /user
- Método: PUT



2.3 Schedule Appointment

Cria um novo appointment, sendo necessário fornecer o id do doctor, hora de início e fim, número da sala, o id do patient e os ids das nurses associadas. Somente os assistants podem realizar esta operação, desse modo também é fornecido o 'token'.

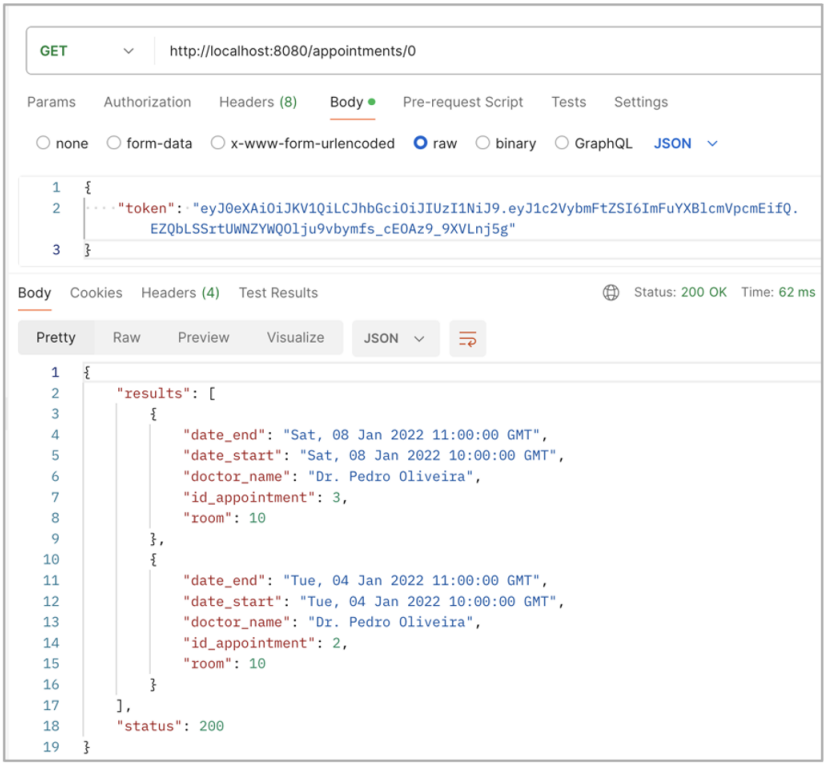
- URL: /appointment
- Método: POST



2.4 See Appointments

Enumera os appointments, com toda informação dos mesmos, de um patient em específico. Apenas assistants e o próprio patient conseguem realizar esta operação.

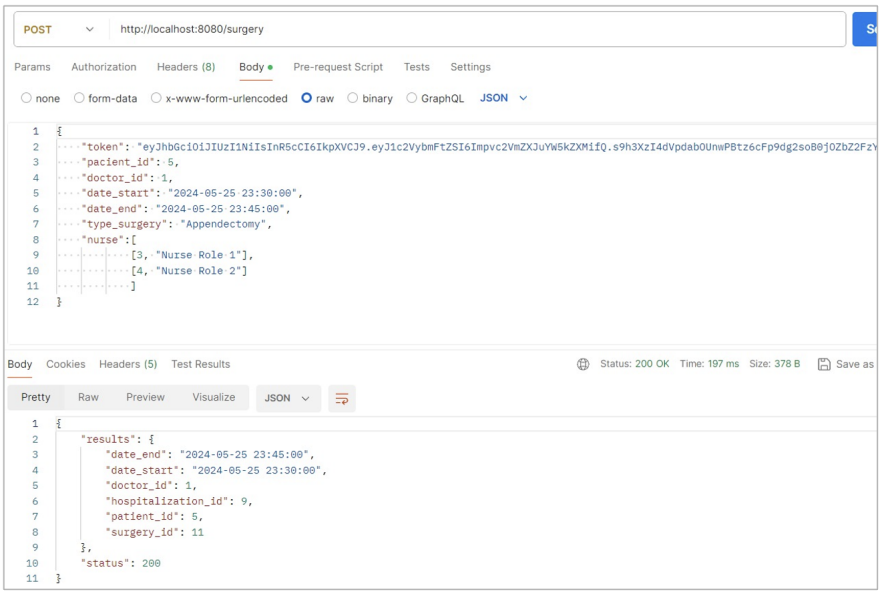
- URL: /appointments/<patient_user_id>
- Método: GET



2.5 Schedule Surgery

Para agendar uma surgery é necessário fornecer: o id do patient, do doctor, das nurses associadas, o tipo de surgery, o início e o fim da mesma. Caso seja fornecido o id de uma hospitalization prévia do patient, estas são associadas. Em ambos os casos é sempre ativo um *'trigger'* que gera uma billing para o patient pagar. Apenas assistants podem executar esta operação, sendo necessário o fornecimento do 'token' para confirmação.

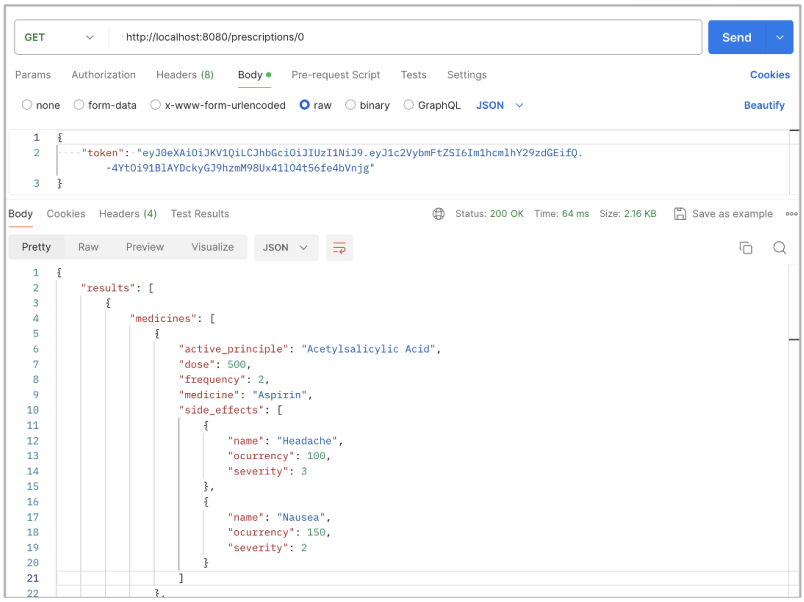
- URL: /surgery/<hospitalization_id>
- URL: /surgery
- Método: POST



2.6 Get Prescriptions

Para visualizar as prescriptions, com os medicines receitados e respetivos side_effects, associadas a cada patient é necessário fornecer o id do mesmo. Todos os employees e o próprio patient podem utilizar este comando, para tal é necessário fornecer o 'token' para verificar esta restrição.

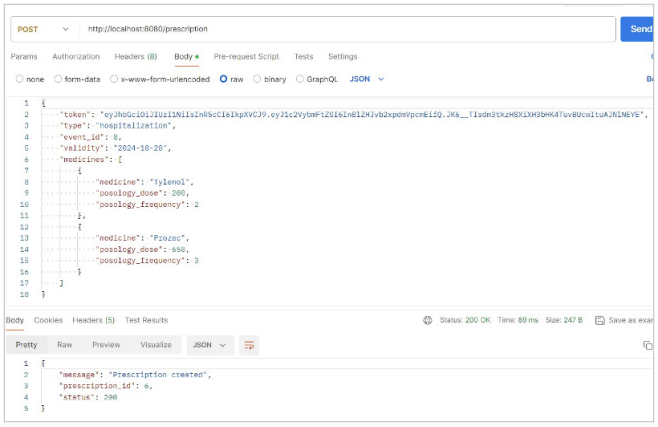
- URL: /prescriptions/<person_id>
- Método: GET



2.7 Add Prescriptions

Quando se realiza uma surgery ou appointment, pode ser necessário atribuir uma prescription, para tal o doctor, é o único cargo com permissões para tal, fornece: o seu 'token', o tipo de atividade que estava a ser realizada (surgery ou appointment), o id do evento, a validade da prescription e os nomes dos medicines, com a respetiva posology_dose e posology_frequecy. Estes medicines já tem que existir na base de dados e tem associados side_effects, que tem uma ocorrency e severity.

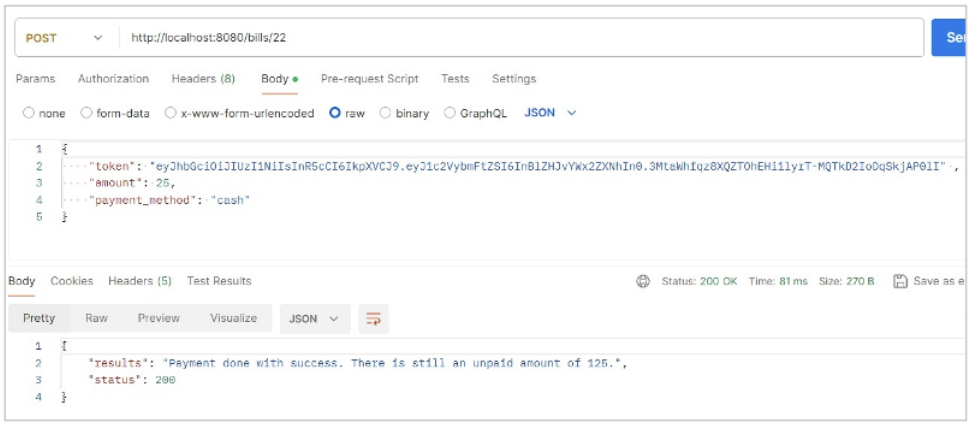
- URL: /precription
- Método: POST



2.8 Execute Payment

Para efetuar o pagamento da billing, o patient tem de fornecer o id associado a esta, tal como o seu 'token', pois só o próprio o pode fazer. Caso o pagamento não seja do valor total da billing, é gerado um payment do valor pago e é descontado na billing correspondente, ficando assim guardado o histórico de todos os payments.

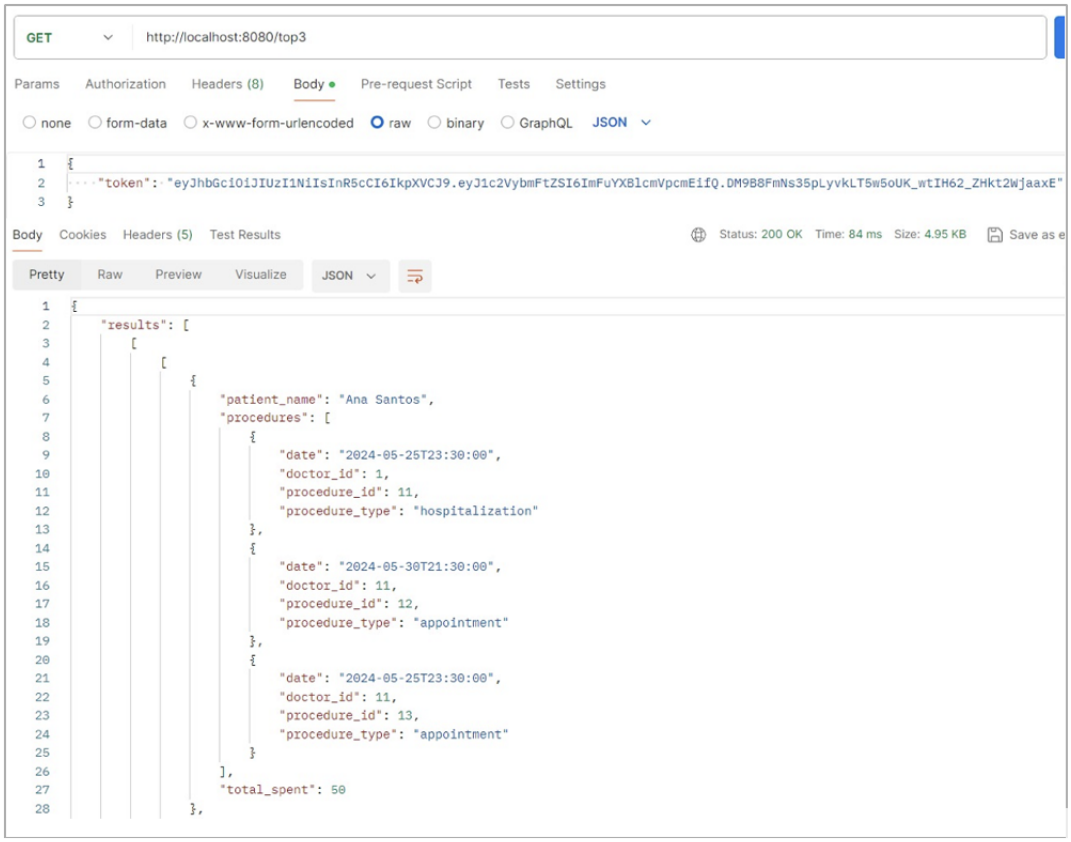
- URL: /bills/<bill_id>
- Método: POST



2.9 List Top 3 patients

Visualização dos 3 patients com mais gastos no hospital no mês atual, qualquer pessoa pode acessar esta função, não sendo necessário fornecer o 'token'.

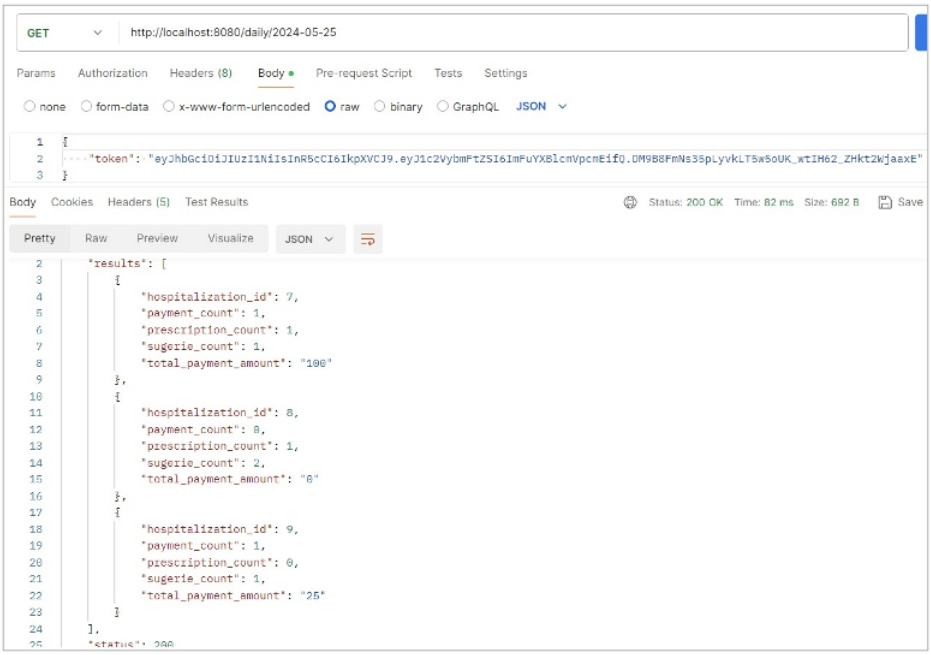
- URL: /top3
- Método: GET



2.10 Daily Summary

Catalogar todas as hospitalizations realizadas no próprio dia. Contêm a informação do número de surgeries, payments e prescriptions. Só os assistants conseguem ter acesso a esta informação, sendo preciso o 'token'.

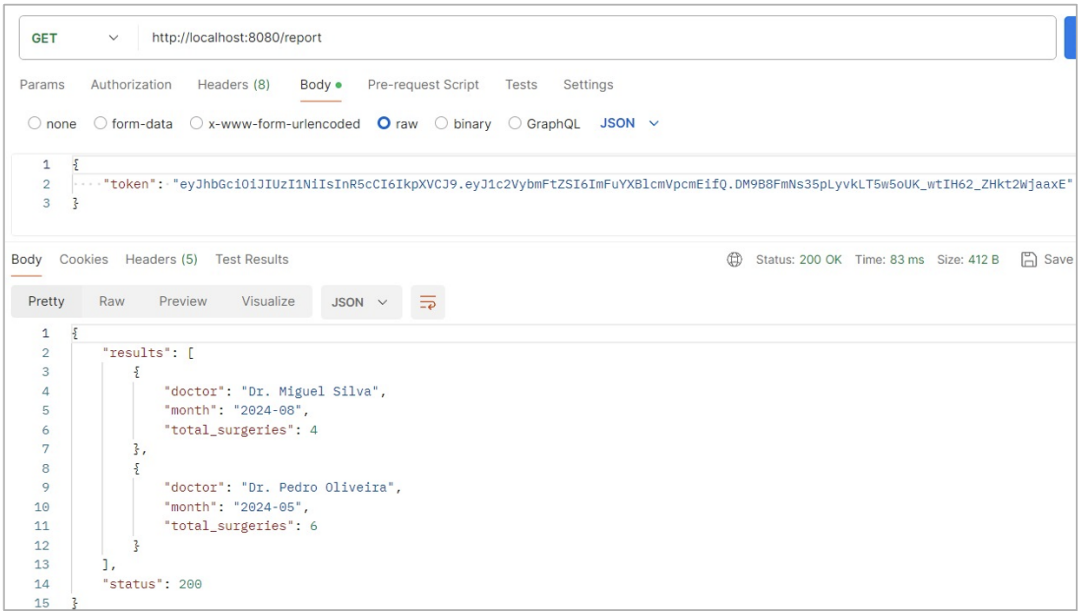
- URL: /daily/<year-month-day>
- Método: GET



2.11 Generate a monthly report

Exibição dos doctors com um maior número de surgeries nos últimos 12 meses, agrupadas mensalmente. É imprescindível fornecer o 'token', uma vez que apenas assistants têm permissão de acesso.

- URL: /report
- Método: GET



3 Additional Information

3.1 Triggers

Além dos códigos-fontes, foi entregue um código em *SQL* que contém os vários *triggers* implemetados neste trabalho. Assim, foram desenvolvidos dois *triggers* :

- **create_billing_on_appointment()** - Este *trigger* é ativo sempre que um registo é inserido na tabela *appointment*. Este cria uma billing com um valor total de 50 euros. Além disso, o valor do campo *status* é inicializado como **True**, indicando que a billing ainda não foi paga. Por fim, os valores são inseridos na tabela *billing*.
- **create_billing_on_hospitalization()** - Semelhante ao *trigger* anterior, este é ativo sempre que um registo for inserido na tabela *hospitalization*. Ele cria uma billing, sendo a única diferença o preço da hospitalization, que custa 150 euros.

3.2 Problemas de Concorrência

Para resolver problemas de concorrência dentro da base de dados foram implementadas algumas estratégias:

3.2.1 Marcação de surgeries e appointments

Para evitar que appointments sejam agendados para o mesmo horário com o mesmo doctor, é ativado um bloqueio (lock) nas tabelas *surgeries* e *appointment* quando a ação é realizada. Dessa forma, assegura-se que não sejam marcadas surgeries ou appointments para o mesmo horário com o mesmo doctor.

3.2.2 Informação relativa às billings

Para evitar que valores sejam corrompidos quando dois payments são realizados simultaneamente para a mesma billing, a linha correspondente na tabela de billings é bloqueada sempre que um valor seja modificado. Isso evita a perda de dados.

3.2.3 Leitura de dados sensíveis

Ao acessar dados que podem ser modificados, é necessário definir **SET TRANSACTION ISOLATION LEVEL REPEATABLE READ** para ignorar quaisquer modificações existentes na base de dados enquanto a query está a ser executada.

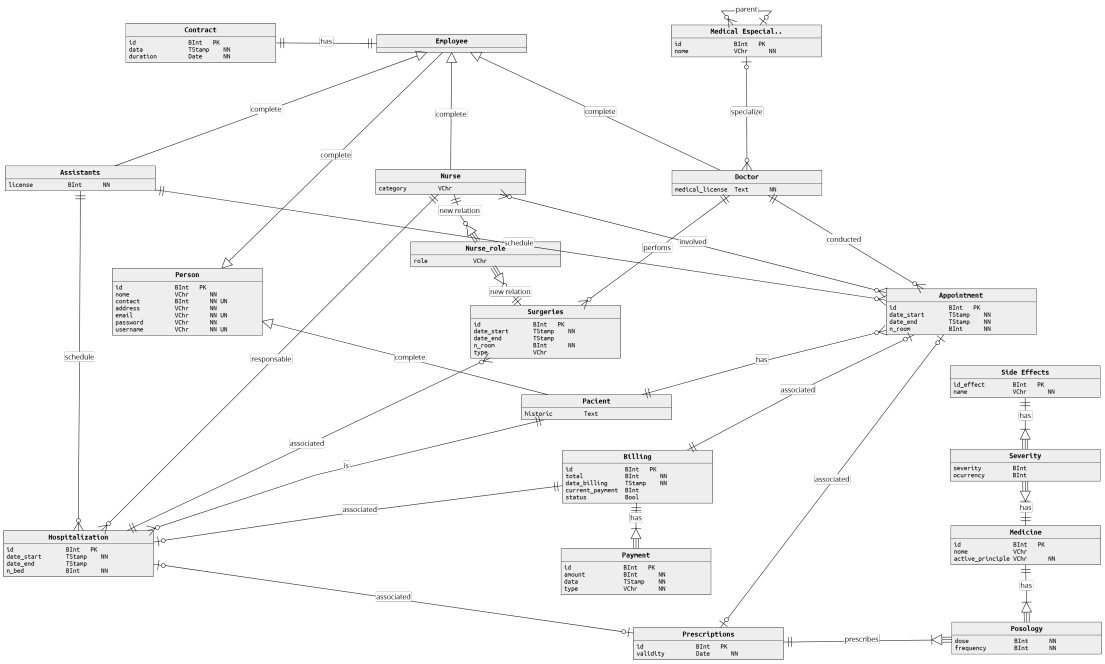
3.3 Divisão do trabalho

Para este projeto foi usada uma divisão inicial do trabalho para uma melhor gestão do tempo:

Tiago	Rui	Mariana
Add users	User Authentication	See Appointments
List Top 3 patients	Schedule Appointment	Add Prescriptions.
Daily Summary	Schedule Surgery	Get Prescriptions
Generate a monthly report	Execute Payment	Relatório e Apresentação

Apesar deste diagrama inicial, todos os membros estiveram presentes ativamente na organização e execução de todos os pontos apresentandos. Foram despendidas 75h no total.

4 Diagrama Entidade-Relacionamento



5 Diagrama Modelo-Relacional