Documentação do Projeto .NET – Grupo 49

Documentação do Projeto *Health Med*

Introdução

Vamos apresentar a documentação do projeto .NET *Health Med*, no qual será apresentado o diagrama de classe da API e diagrama dos *endpoints*.

Diagrama de Classe

As imagens dos diagramas de classes abaixo é uma representação visual das classes, seus atributos, e as relações entre elas. Ele fornece uma visão clara da estrutura do projeto e ajuda a entender como as diferentes partes do sistema se conectam.

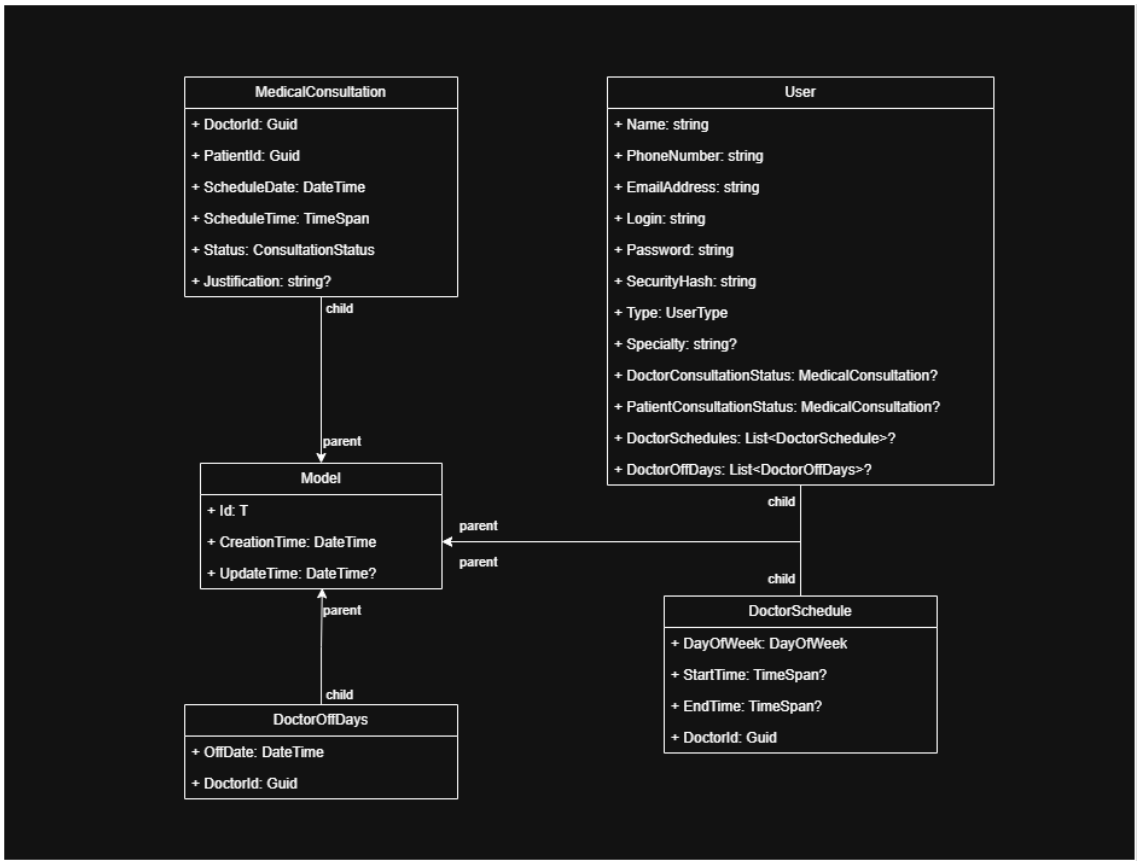

Figura 1 - Diagrama de classe (Hereditariedade)

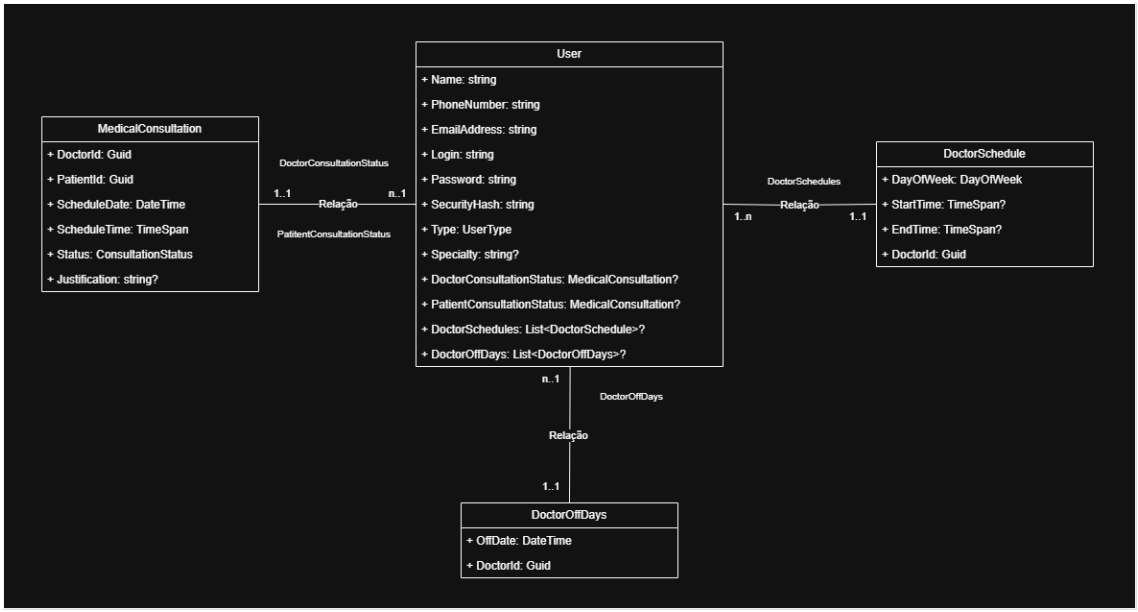

Figura 2 - Diagrama de classe (Relação das classes)

Classes Principais

- **Classe *Model***: Esta é a classe raiz de todas as outras classes presente no sistema. Contém os atributos comum entre todas as classes, como Id, *CreationTime* (Data de criação) e *UpdateTime* (Data de atualização, onde salvamos a última alteração).
- **Classe *User***: Representa os usuários do sistema, como Paciente e Médico. Esta classe é filha da classe *Model*. Inclui todos os atributos de *Model*, e atributos como *Name* (Nome), *PhoneNumber* (Telefone), *EmailAddress* (Endereço de Email), *Login* (Usuario de autenticação), Password (Senha), *SecurityHash* (Hash de segurança para validação da senha), *Type* (Tipo de usuário), *Specialty* (Caso tenha alguma especialidade), e variáveis de relação como *DoctorConsultationStatus, PatientConsultationStatus, DoctorSchedules, DoctorOffDays*.
- **Classe *MedicalConsultation***: Define as consultas marcadas e a relação entre o médico e paciente na consulta. Esta classe é filha da classe *Model*. Inclui todos os atributos de *Model, e c*ada *MedicalConsultation* possui atributos como *DoctorId* (Identificador do médico), *PatientId* (Identificador do paciente), *ScheduleDate* (Data da consulta), *ScheduleTime*(Hora da consulta), *Status*, *Justification* (Justificativa caso seja cancelada a consulta).
- **Classe *DoctorSchedule***: Representa o calendário de consultas do médico. Esta classe é filha da classe *Model*. Inclui todos os atributos de *Model*, e inclui atributos como *DayOfWeek* (dia da semana), *StartTime* (Hora da consulta), *EndTime* (Hora que finaliza a consulta) e *DoctorId*(Identificador do médico).

- **Classe *DoctorOffDays***: Representa o calendário de dias em que o médico está indisponível. Esta classe é filha da classe *Model*. Inclui todos os atributos de *Model*, e inclui atributos como *OffDate* (dia em que está indisponível), e *DoctorId*(Identificador do médico).

API do Projeto

A API do projeto .NET permite a interação programática com o sistema. Ela oferece endpoints para realizar operações CRUD (Create, Read, Update, Delete) nas diversas entidades do sistema.

O projeto foi dividido em dois microserviços, no qual um é focado na criação e atualização dos dados no banco de dados e outro de fazer GET de dados e autenticação.

No micro-serviço HealthMed.CommandAPI, temos:

- Criar um paciente/médico.
- Criação de:
    - Consulta médica.
    - Datas disponíveis para consulta.
    - Dias indisponíveis no calendário.
- Atualização de:
    - Consulta (Atualizar alguma informação e/ou cancelar uma consulta).

No micro-serviço HealthMed.QueryAPI, temos:

- Métodos de autenticação para médico e paciente.
- Métodos de listagem de:
    - Médicos.
    - Calendário médico.
    - Consultas.
    - Consultas pendentes.

Endpoints Disponíveis

Autenticação

- POST /auth/login: Faz a autenticação do usuário.

Médico

- GET /doctor/pendingConsultations?PageNumber=1: Retorna uma lista de todas as consultas pendentes do médico.

- GET /doctor/consultations?PageNumber=1: Retorna uma lista com todas as consultas do médico.
- GET /doctor?PageNumber=1: Rretorna uma lista com todos os médicos disponíveis.
- GET /doctor/{id}: Retorna uma lista de datas/horários disponíveis do médico indicado.
- POST /user/createMedicUser: Criar um usuário do tipo médico.
- POST /doctor/updateConsultation: Atualizar uma consulta.
- POST /doctor/createSchedule: Cria uma data e hora disponível para consulta.
- POST /doctor/createOffDay: Cria uma data indisponível para consulta.

Paciente

- GET /patient/consultations?PageNumber=1: Retorna uma lista com todas as consultas do paciente.
- POST /patient/scheduleConsultation: Cria a data e hora da consulta.
- POST /patient/cancelConsultation: Cancela uma consulta agendada.

Soluções escolhidas

Optamos por utilizar micro-serviços para flexibilizar e isolar os tipos de ações da API, assim utilizando um para criação e atualização de dados no banco de dados e outro para as solicitações e autenticação.

Como banco de dados optamos por utilizar o SQL Server, sendo um banco de dados amplamente utilizado pelo mercado e estruturado. Para as requisições de informações no banco de dados utilizamos o Dapper, por dar uma flexibilidade para as requisições.

Para a parte de monitoramento e *logs* escolhemos utilizar Serilog para geração de logs e para o monitoramento da API Prometheus e Grafana.

Para os testes unitários utilizamos XUnit.

Escolhemos utilizar o API Gateway para facilitar os gerenciamentos de todos os micro-serviços, para que o versionamento e os testes.

Conclusão

Para esta API optamos por uma arquitetura de micro-serviços para garantir flexibilidade e isolamento das ações da API. Utilizamos SQL Server como banco de dados, Dapper para requisições, Serilog para logs, Prometheus e Grafana para

monitoramento e XUnit para testes unitários. E para o gerenciamento central de todas os micro-serviços, a escolha foi o API Gateway.

Essas escolhas proporcionaram uma solução robusta, escalável e eficiente, atendendo adequadamente às necessidades do projeto.