INSTITUTO POLITÉCNICO
DO CÁVADO E DO AVE
ESCOLA SUPERIOR DE TECNOLOGIA

RELATÓRIO DE TRABALHO PRÁTICO I

Sistema de Gestão em uma Crise de Saúde Pública

LUCAS BRGA MENDONÇA

ALUNO Nº 17870

Trabalho realizado sob a orientação de:
Luís Ferreira

Linguagens de Programação II

Licenciatura em Engenharia de Sistemas Informáticos

Barcelos, Abril de 2020

Índice

Lista de Figuras

1 Introdução

O presente trabalho tem como objetivo principal construir um sistema de gestão com os paradigmas das linguagens orientadas a objeto, especificamente o C#, que auxilie de alguma maneira numa crise de saúde pública. Ferramentas na gestão de pessoas infectadas serão desenvolvidas ao longo do projeto, tais como registro de novos casos, contabilização total de casos por região, sexo, faixa etária, dentre outros que são úteis no mesmo âmbito.

Para a primeira entrega do sistema, foram estruturadas as principais classes, assim como o relacionamento entre estas mesmas classes. O código está estruturado de forma que as classes sejam fácilmente identificadas e de forma que utilize alguns dos conceitos fundamentais vistos até hoje nas aulas de Linguagem de Programação II. O presente relatório, tal como o código completo do projeto está disponível no GitHub, através do link 17870_LP2.git.

2 Estrutura do projeto

Para uma melhor disposição e estruturação dos dados, as principais classes do programa foram divididas em três pastas até o momento: *Entities*, *Models e Interfaces*.

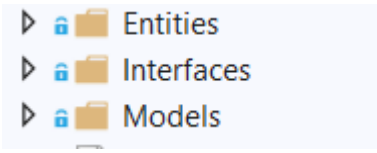

Figura 1: Pastas do projeto

a) Models

Contém as classes responsáveis por manter os dados da aplicação. Pode, num exemplo concreto, recuperar e armazenar o estado do modelo em um servidor de banco de dados.

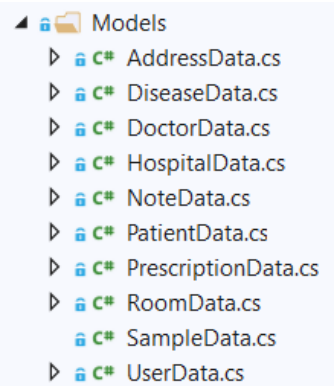

Figura 2: Classes da pasta Models

b) Interfaces

Contém as interfaces que podem ser implementadas nas classes derivadas. As interfaces possuem métodos comuns a maioria das classes existentes no sistema.

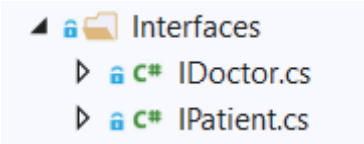

Figura 3: Classes da pasta Interfaces

c) Entities

Contém as classes responsáveis por gerir a camada de dados. Contém as regras de negócio e as operações que podem ser realizadas dentro do sistema.

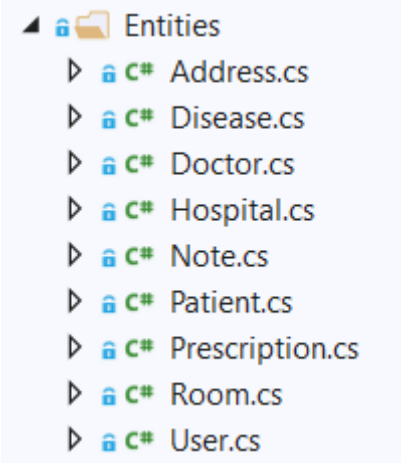

Figura 4: Classes da pasta Entities

O diagrama de classes em sua totalidade com todas as suas relações pode ser visto na figura 5.

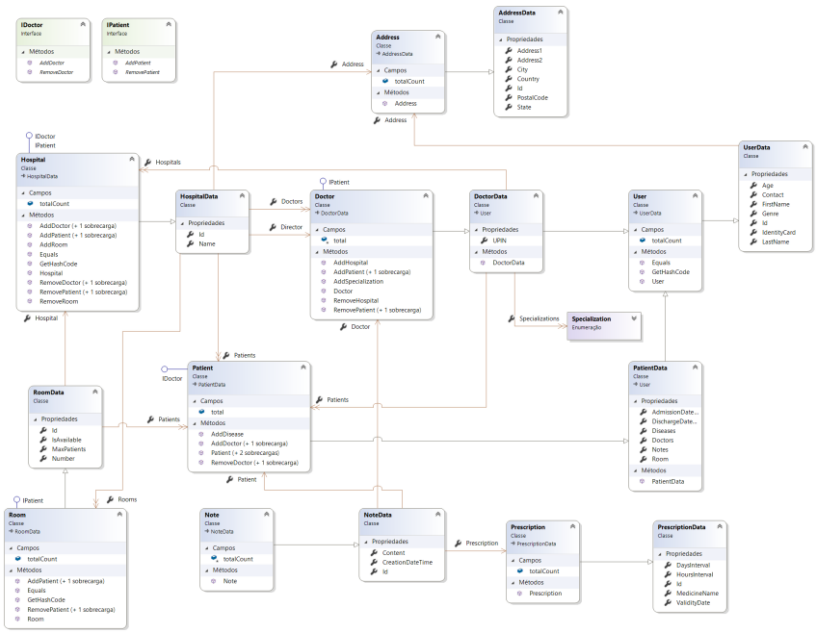

Figura 5: Diagrama de classes do sistema

2.1 Descrição das classes

a) *User.cs*

Classe responsável por gerir dados de um usuário. É uma classe geral utilizada como pai de outras classes dentro do sistema. Herda o modelo de dados de UserData.cs.

b) *Patient.cs*

Classe responsável por gerir dados de um paciente. Herda o modelo de dados de PatientData.cs, que por sua vez herda de User.cs.

c) *Doctor.cs*

Classe responsável por gerir dados de um médico. Herda o modelo de dados de DoctorData.cs, que por sua vez herda de User.cs.

d) *Address.cs*

Classe responsável por gerir os endereços de usuários cadastrados no sistema, sejam eles pacientes ou médicos, além de gerir os endereços de hospitais. Herda modelo de dados de AddressData.cs

e) *Disease.cs*

Classe responsável por gerir as doenças dos pacientes de um hospital. Herda modelo de dados de DiseaseData.cs.

f) *Hospital.cs*

Classe responsável por gerir hospitais. Herda modelo de dados de HospitalData.cs.

g) *Note.cs*

Classe responsável por gerir a ficha médica de um paciente. É nesta classe que são encontradas informações relativas ao diagnóstico do doente, observações e preescrições médicas. Herda modelo de dados de NoteData.cs

h) *Prescription.cs*

Classe responsável por gerir uma receita associada a uma nota (ficha médica). É nesta classe que estão dispostas informações sobre o medicamento receitado, assim como seu intervalo de dias e horas e prazo de válidade da mesma. Herda modelo de dados de PrescriptionData.cs.

i) *Room.cs*

Classe responsável por gerir os quartos disponíveis em um hospital. Herda modelo de dados de RoomData.cs.

j) *Enumeração: Specialization.cs*

Contém as especializações médicas disponíveis no sistema.

2.2 Declaração de atributos de associação

Algumas propriedades de relacionamento foram marcadas como virtuais, pois é assim que o padrão Entity Framework pode implementar o chamado “carregamento lento” de forma quase que automática. Este carregamento significa que os dados relacionados são carregados de modo transparente do banco de dados quando a propriedade de navegação é acessada. Dado um contexto de acesso ao banco de dados, a propriedade que possui a relação só carregaria seus elementos se fosse realmente usada.

Embora o padrão Entity Framework e o carregamento lento não sejam utilizados no sistema até o momento, a escolha de seguir desenvolver de acordo com esses padrões é uma forma de praticar os conceitos até o cenário hipotético tornar-se realmente necessário.

3 Conclusão

Conclui-se que até a presente entrega 1, a estrutura básica do sistema esteja em funcionamento, mas não impossibilitando o código de sofrer alterações e melhorias ao longo de todo o projeto. As Interfaces e classes abstratas serão melhores exploradas de forma que os as ações comuns a maioria das classes utilizadas estejam centralizados nestes objetos.

Bibliografia

Anexos