

DIRETORIA: Diretoria Acadêmica de Gestão e Tecnologia da Informação (DIATINF)

DISCIPLINA: Programação Orientada a Objetos - POO - 2023.2

**DOCENTE: Gilbert Azevedo da Silva** 

**ALUNO: Felipe da Costa Ferreira** 

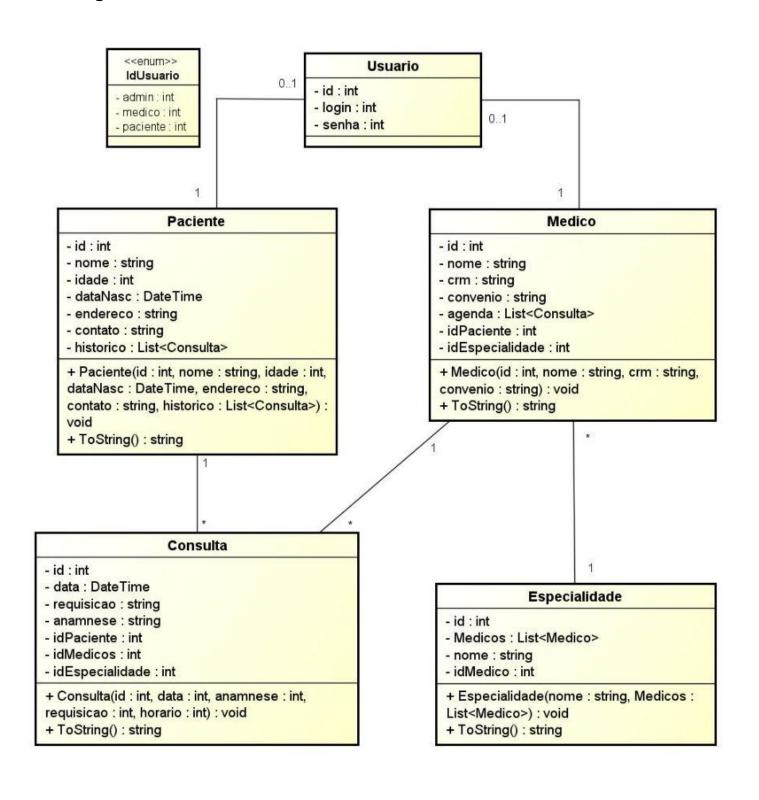
Modelagem de Sistema Clínica Médica

### 1. Descrição do sistema

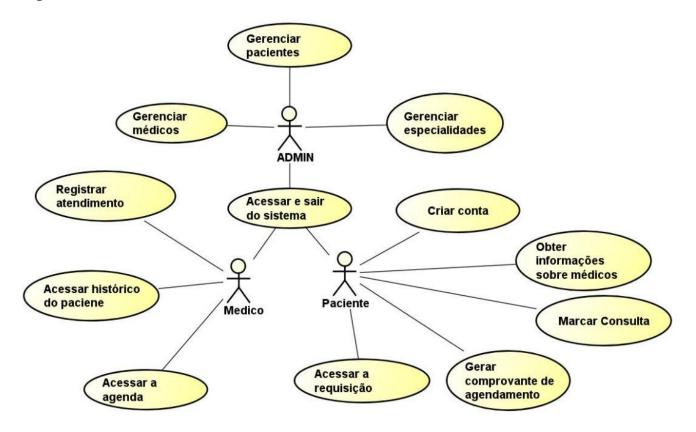
Uma clínica médica solicitou o desenvolvimento de um protótipo de sistema para melhorar o atendimento fornecido. A proposta inicial é que o software permita que um administrador possa gerenciar especialidades, médicos e pacientes. Os profissionais poderão checar sua agenda do dia, gerenciar o histórico do paciente e as informações sobre eles (nome, especialidade) possam estar ao alcance do usuário antes de optar por ser paciente (realizar cadastro). Por fim, o paciente deve ser capaz de marcar a consulta, emitir o comprovante e também a última requisição passada pelo médico.

Abaixo está a modelagem desse sistema proposto pelo médico.

## 2. Diagrama de classes - Entidades



# 3. Diagrama de classes - Casos de Uso



### 4. Diagrama de classes - Operações

#### View

#### <<static>>

#### View

- + PacienteInserir(nome : string, idade : int, dataNasc : DateTime, encedereco
- : string, contato : string) : void
- + PacienteListar() : List<Paciente>
- + PacienteExcluir(id : int) : void
- + PacienteAtualizar(id : int, nome : string, idade : int, dataNasc : DateTime, encedereco : string, contato : string) : void
- + MedicoInserir(nome : string, crm : string, especialidade : string, convenio : string) : void
- + MedicoListar(): List<Medico>
- + MedicoExcluir(): void
- + MedicoAtualizar(nome : string, crm : string, especialidade : string, convenio : string, id : int) : void
- + EspecialidadeInserir(nome : string, listaMed : List<Medico>) : void
- + EspecialidadeListar() : List<Especialidade>
- + EspecialidadeAtualizar(nome: string, listaMed: List<Medico>, id:int): void
- + EspecialidadeExcluir(id : int) : void
- + ConsultaInserir(data: DateTime, anamnese: string, requisicao: int): void
- + ConsultaListar(): List<Consulta>
- + ConsultaAtualizar(id : int, data : DateTime, anamnese : string, requisicao : string) : void
- + ConsultaExcluir(id: int): void
- + MenuCadastro(): void
- + InserirDadosPaciente(nome : string, idade : int, dataNasc : DateTime,
- endereco: string, contato: string): string
- + MostrarEspecialidades() : List<Especialidade>
- + MostrarMedicos(): Especialidade
- + MostrarDisponibilidade(): List<AgendaMedico>
- + ConfirmarConsulta(): void
- + ImprimeComprovante(): string
- + AbrirRequisicao(): Paciente
- + ImprimeRequisicao(): void
- + AtualizarHistorico(obj : Paciente) : void
- + AbrirHistorico(paciente : Pacente) : List<Consulta>
- + AbrirMinhaAgenda(): List<AgendaMedico>
- + IniciarConsulta(): void
- + EncerrarAtendimento(): void
- + InserirNovoMedico(): void
- + AtualizarMedicos(): void
- + ExcluirMedico(): void
- + ChecarMedicos(): void

## 5. Diagrama de classes - Classes de Persistência

## << NPaciente >>

- pacientes : List<Paciente>
- + InserirPaciente(): void
- + ListarPaciente(): void
- + AtualizarPaciente(): void
- + ExcluirPaciente(): void

### << NConsulta >>

- consultas : int
- + InserirConsulta(): void
- + ListarConsulta(): void
- + ExcluirConsulta(): void
- + AtualizarConsulta(): void

## << NMedico >>

- medicos : List<Medico>
- + InserirMedico(): void
- + ListarMedico(): void
- + ExcluirMedico(): void
- + AtualizarMedico(): void

## << NEspecialidade >>

- especialidades : List<Especialidade>
- + InserirEspecialidade(): void
- + ListarEspecialidade(): void
- + ExcluirEspecialidade(): void
- + AtualizarEspecialidade(): void

## 6. Diagrama de classes - Interface com o Usuário

