



**MEDICLASS**

Sistema de Prontuário Eletrônico e Apoio à Decisão  
Clínica em Python



MEDICLASS

Universidade Federal de Minas Gerais  
Escola de Engenharia  
Departamento de Engenharia Elétrica

ELE078 Programação Orientada a Objetos, Turma TECAD  
Trabalho Prático 2025/1  
Matheus Marcondes de Oliveira (2022112517)

# Documentação

- Relatório descritivo
- Registro de desenvolvimento de software
- Código fonte
- Apresentação



Disponível em [github.com/matheusmarcondes1/mediclass](https://github.com/matheusmarcondes1/mediclass)

# Sistema

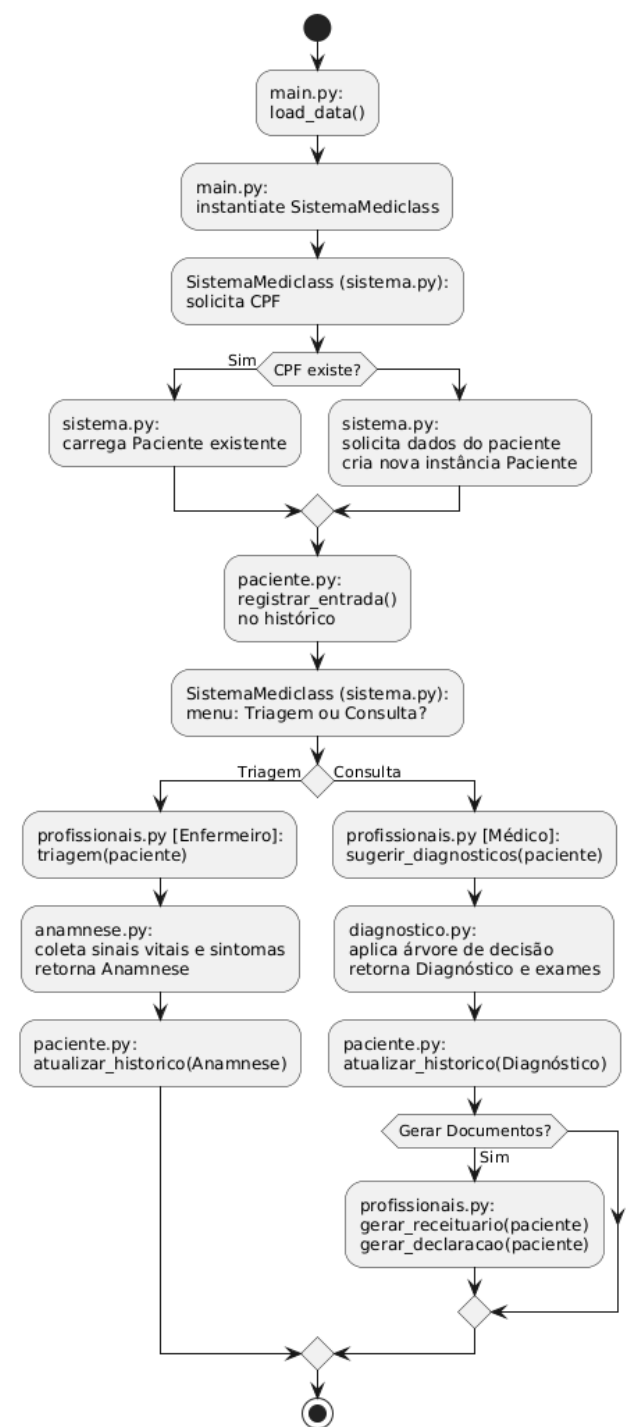
**main.py:** carrega dados salvos, instancia o objeto principal (SistemaMedicclass) e trata a inicialização de usuários padrão.

**sistema.py:** reside a classe que gerencia a funcionalidade de login, exibe o menu principal e despacha chamadas para operações de registro e interação com o usuário.

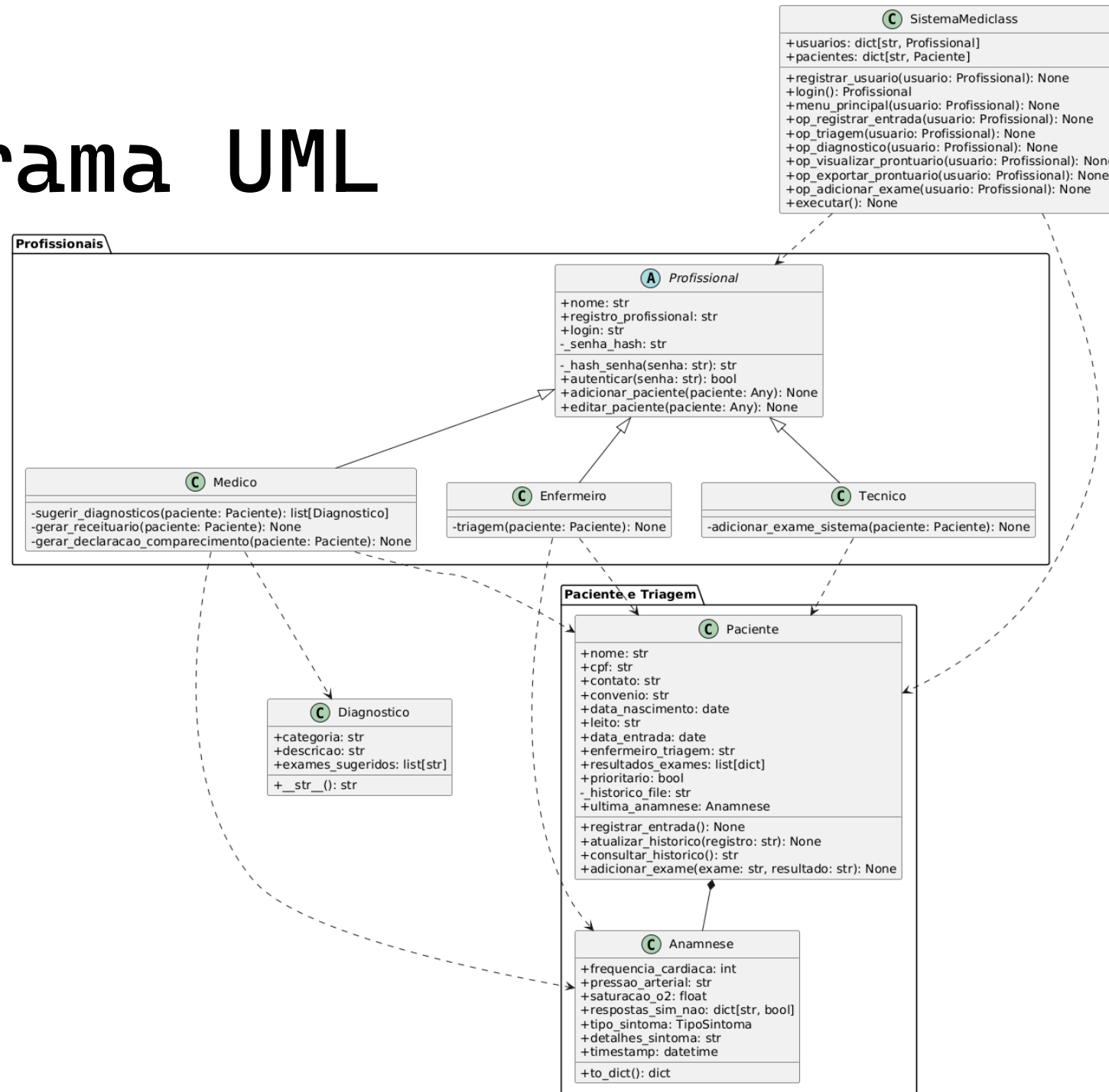
**profissionais.py:** define a hierarquia de usuários, incluindo métodos para autenticação e roteiro de execução de tarefas clínicas pré-definidas para cada subclasse.

**paciente.py** está a classe Paciente, responsável por armazenar dados cadastrais, histórico de atendimentos e gerenciamento de arquivos de entrada do histórico

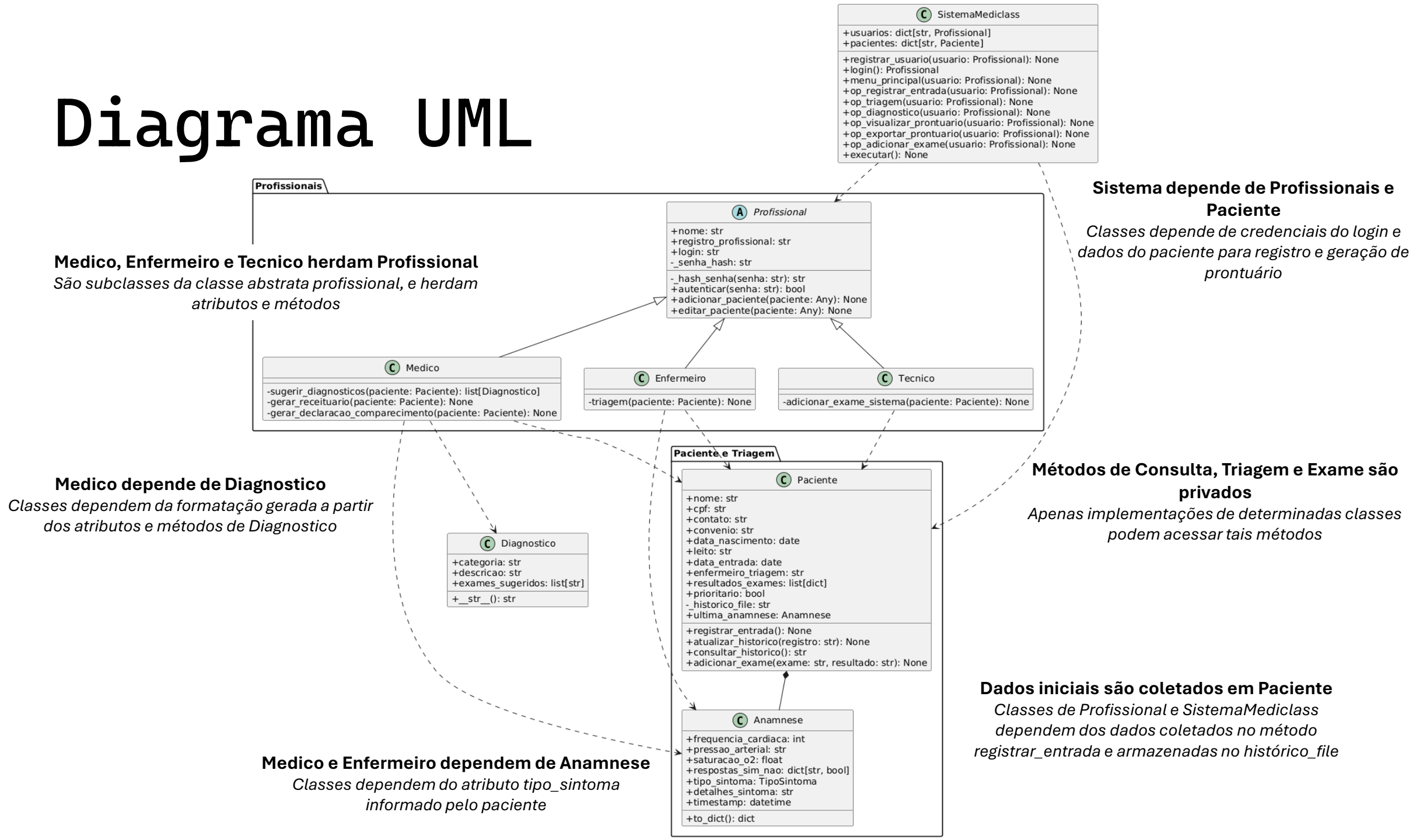
**anamnese.py** e **diagnostico.py:** modelam, respectivamente, a coleta de sinais vitais e sintomas, e a lógica de sugestão de possíveis diagnósticos



# Diagrama UML



# Diagrama UML



# Paciente

## Prontuário exportado do sistema [.txt]

Prontuário de matheus marcondes

CPF: 12345678900

Contato: 31988770000

Convênio: sus

Data de nascimento: 2000-10-10

Leito: 1A

Enfermeiro: Enf. Teste

Prioritário: Sim

Histórico Médico:


Histórico de matheus marcondes (CPF: 12345678900)

Criado em: 2025-07-03 15:36:07

-----  
[2025-07-03 15:36:07] Entrada no leito 1A em 2025-07-03

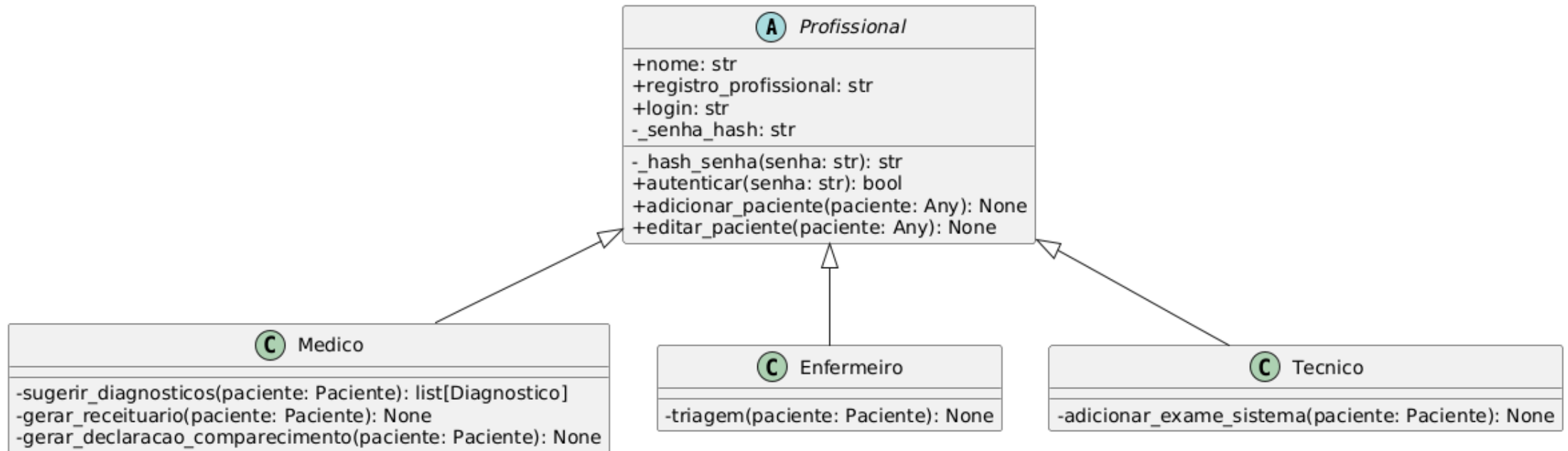
[2025-07-03 15:37:44] Triagem: {'frequencia\_cardiaca': 70, 'pressao\_arterial': '140/100', 'saturacao\_o2': 99, 'respostas\_sim\_nao': {'tosse': True, 'dispneia': True, 'expectoração': True, 'hemoptise': True, 'dor torácica': True, 'febre': True}, 'tipo\_sintoma': 'respiratório', 'detalhes\_sintoma': None, 'timestamp': '2025-07-03 15:37:44'}

[2025-07-03 15:37:44] FLAG: Prioridade ativada devido a valores incomuns.

 Paciente

+nome: str  
+cpf: str  
+contato: str  
+convenio: str  
+data\_nascimento: date  
+leito: str  
+data\_entrada: date  
+enfermeiro\_triagem: str  
+resultados\_exames: list[dict]  
+prioritario: bool  
- \_historico\_file: str  
+ultima\_anamnese: Anamnese  
  
+registrar\_entrada(): None  
+atualizar\_historico(registro: str): None  
+consultar\_historico(): str  
+adicionar\_exame(exame: str, resultado: str): None

# Profissional





# Diagnóstico

Categoria: **Respiratório**

1. Tosse aguda

2. Dispneia

3. Dor torácica pleurítica

Selecione o subgrupo (1-3): **2**

Início súbito com chiado e histórico asmático? (S/N): **s**

--- Diagnósticos sugeridos ---

Respiratório: Possível exacerbação de asma (Exames sugeridos: Espirometria ou PEFR)



Diagnostico

+categoria: str  
+descricao: str  
+exames\_sugeridos: list[str]

+\_\_str\_\_(): str

# Anamnese

Última Anamnese:

frequencia\_cardiaca: 70

pressao\_arterial: 140/100 # concatenação das variáveis de pressão

saturacao\_o2: 99

respostas\_sim\_nao: {'tosse': True, 'dispneia': True, 'expectoração': True, 'hemoptise': True, 'dor torácica': True, 'febre': True}

tipo\_sintoma: respiratório

detalhes\_sintoma: None

timestamp: 2025-07-03 15:37:44

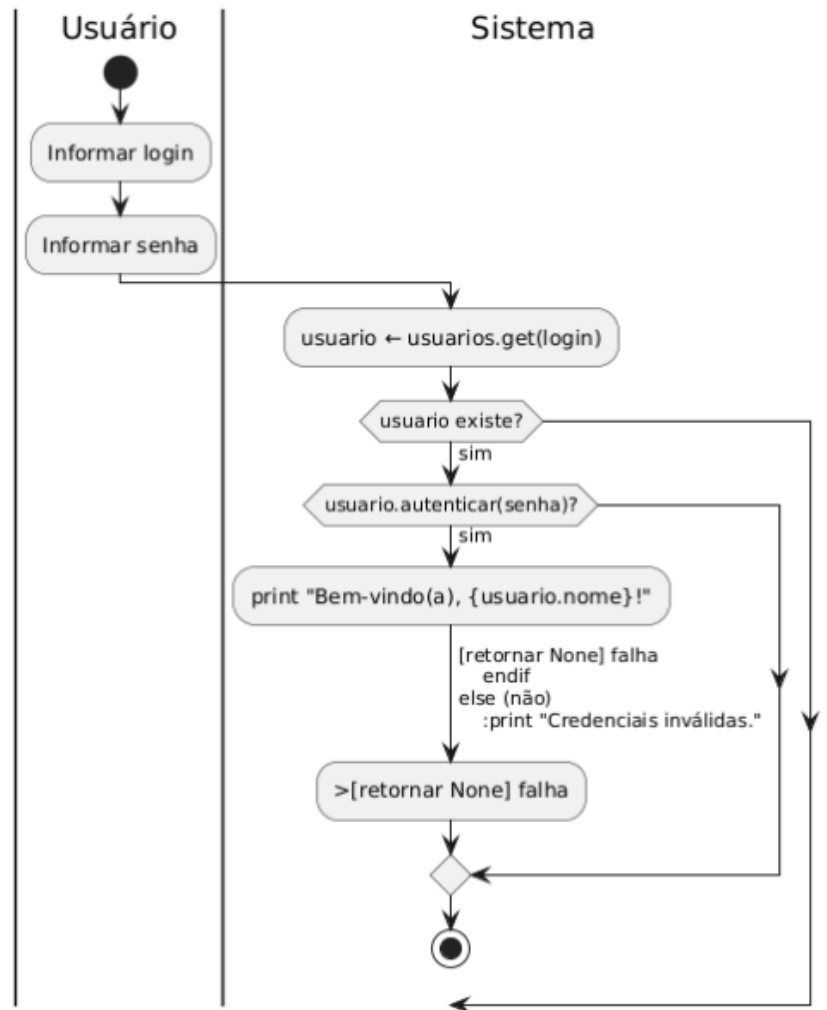


Anamnese

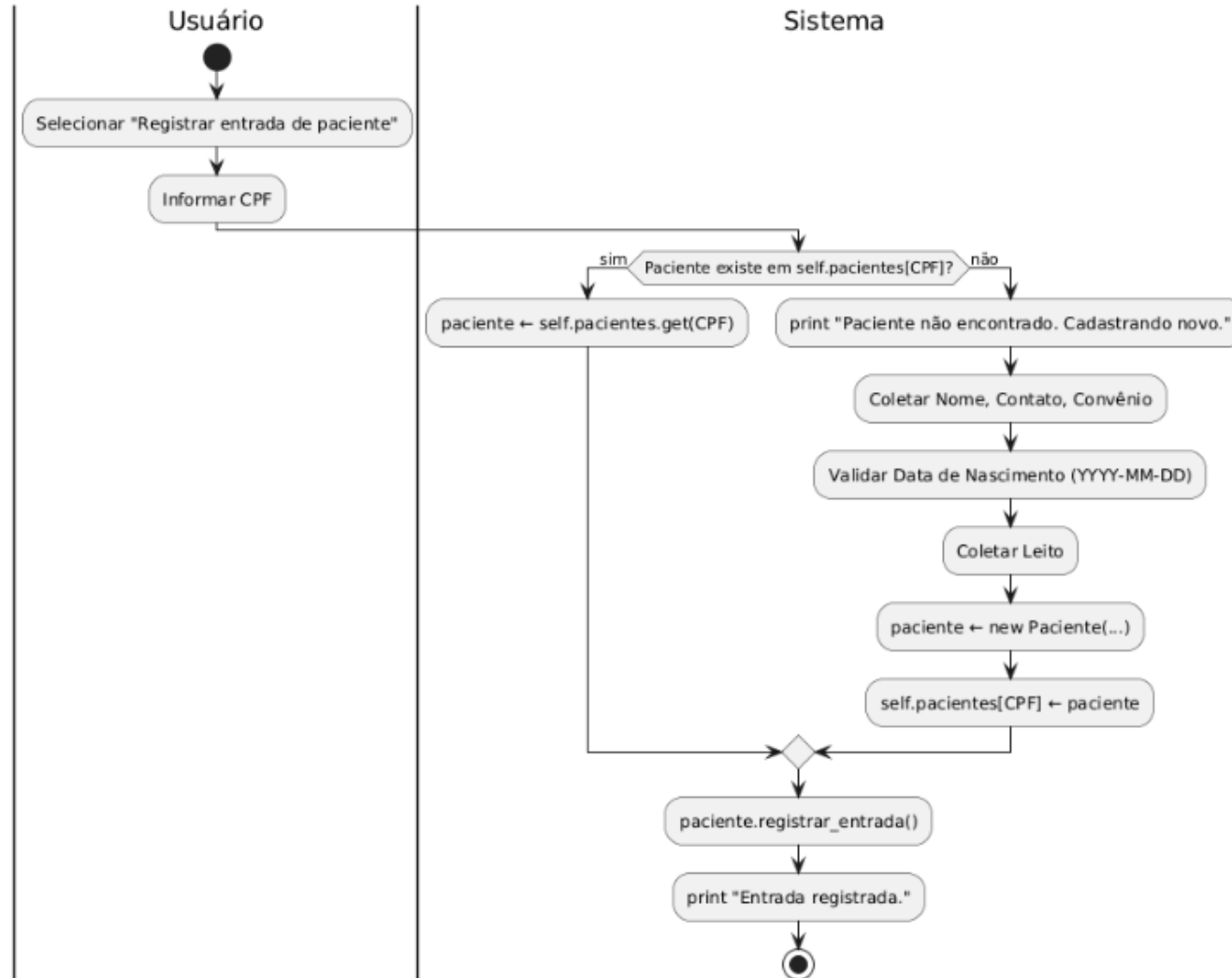
```
+frequencia_cardiaca: int
+pressao_arterial: str
+saturacao_o2: float
+respostas_sim_nao: dict[str, bool]
+tipo_sintoma: TipoSintoma
+detalhes_sintoma: str
+timestamp: datetime

+to_dict(): dict
```

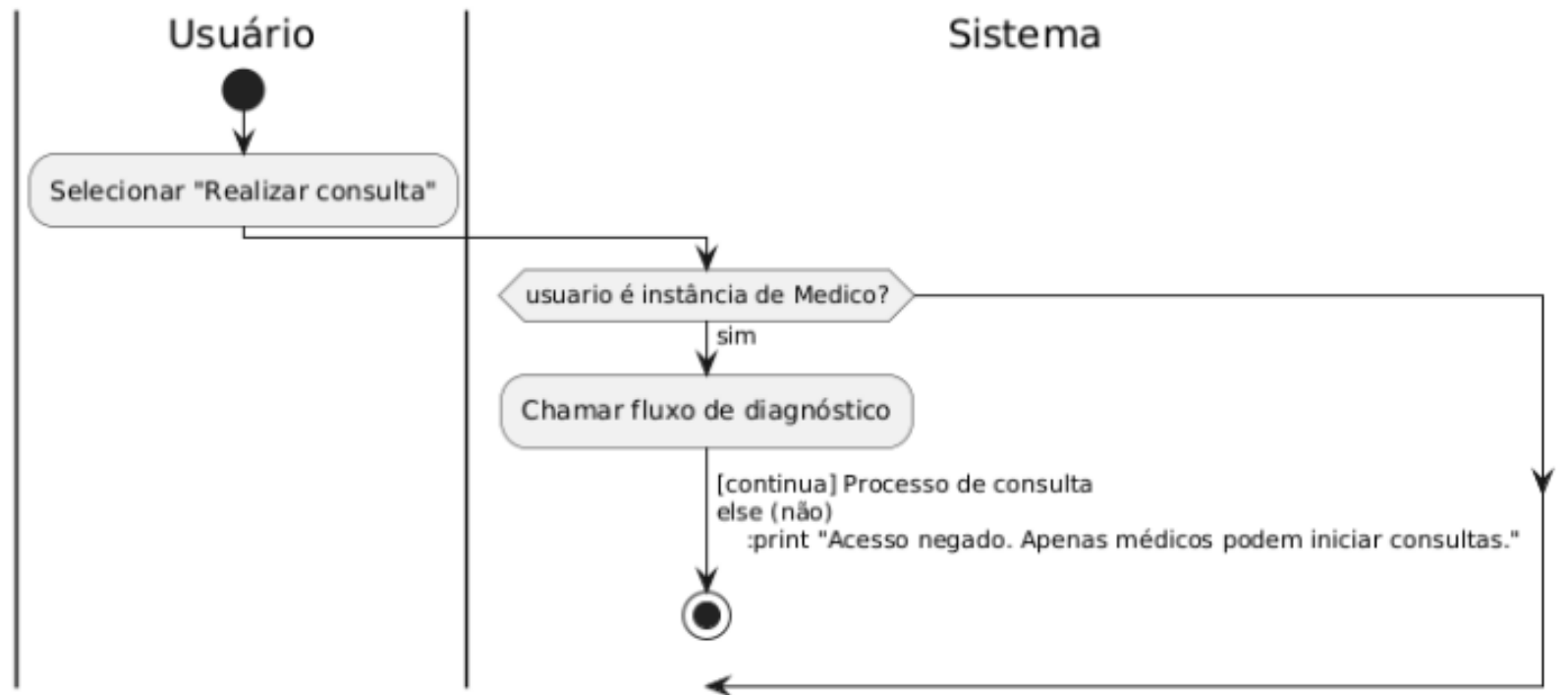
# Processo de autenticação



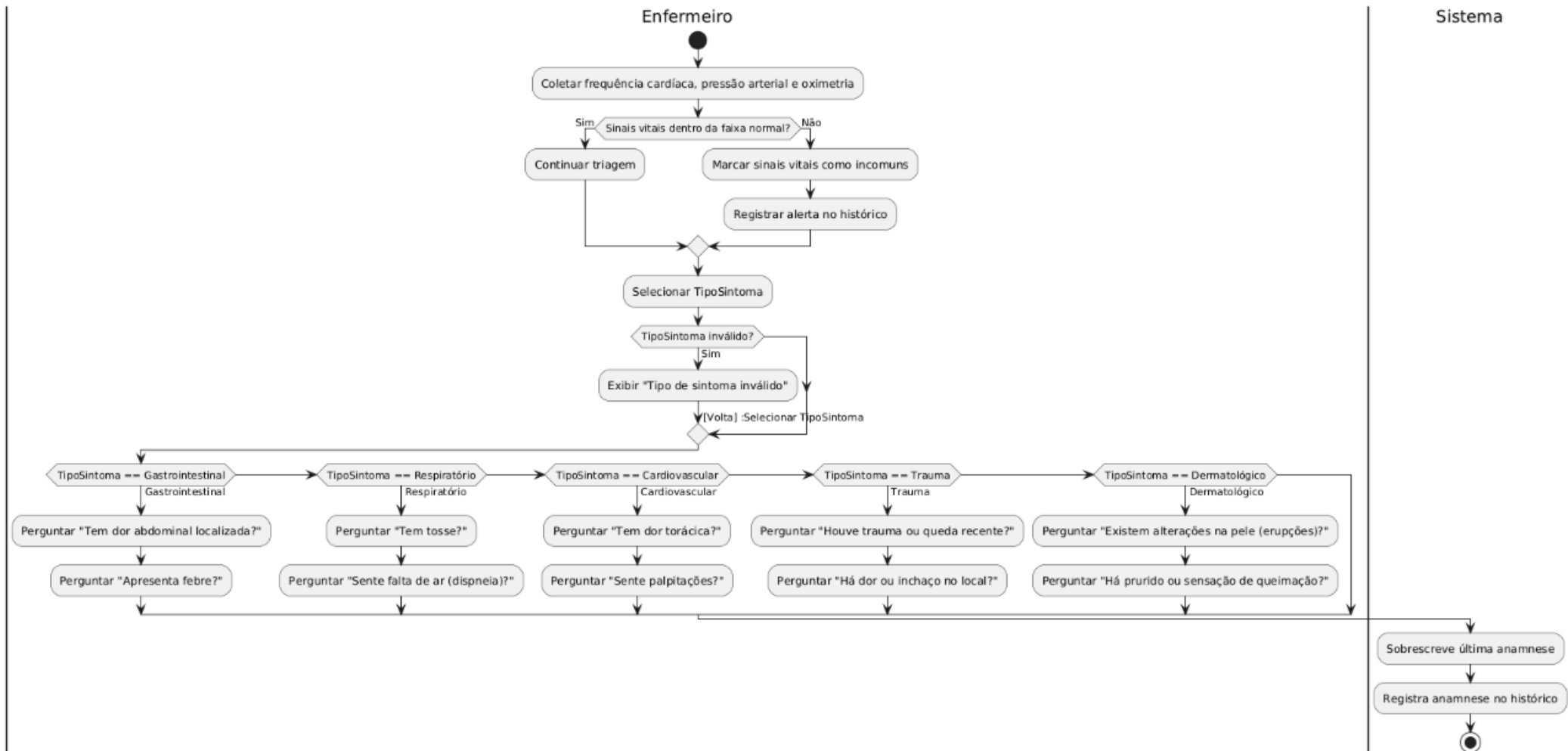
# Registro de paciente



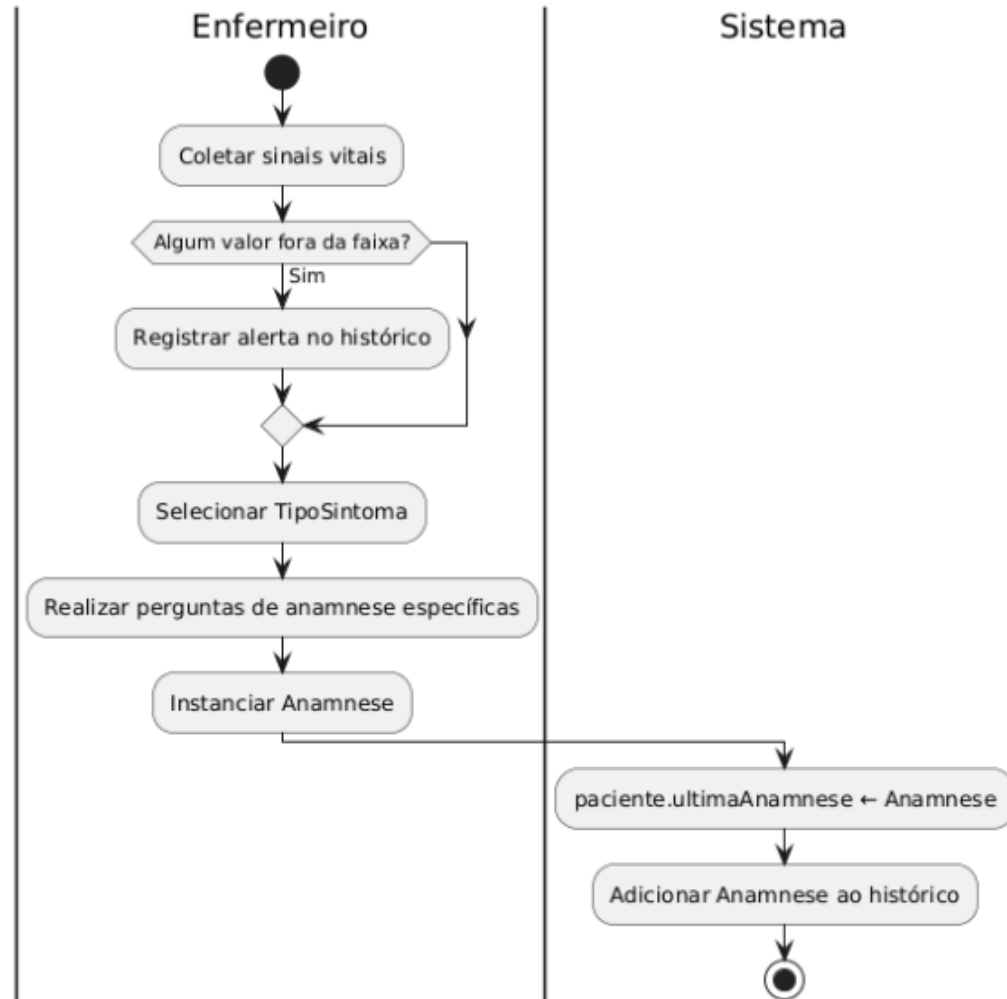
# Verificação de classe para execução de métodos protegidos



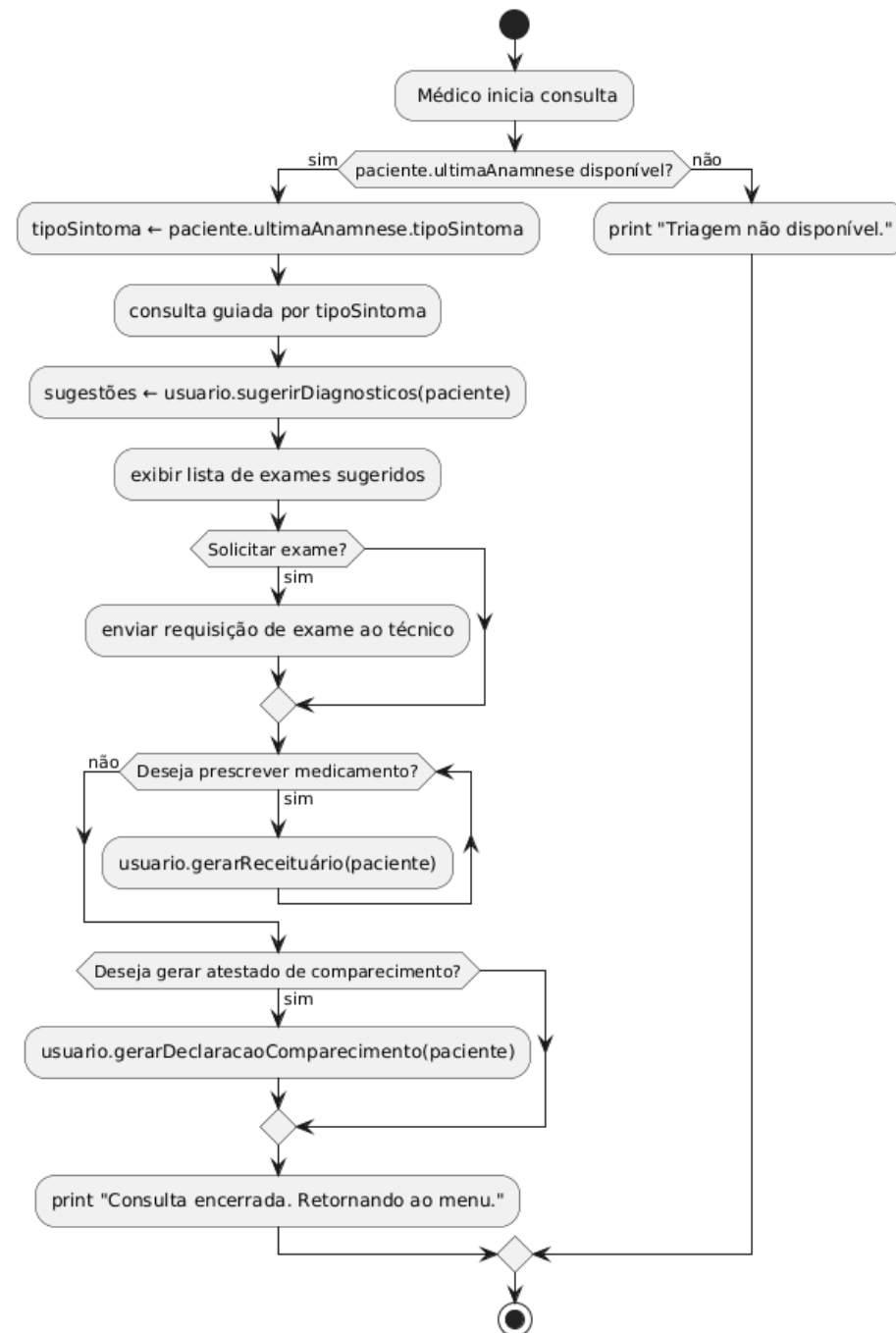
# Triagem



# Triagem

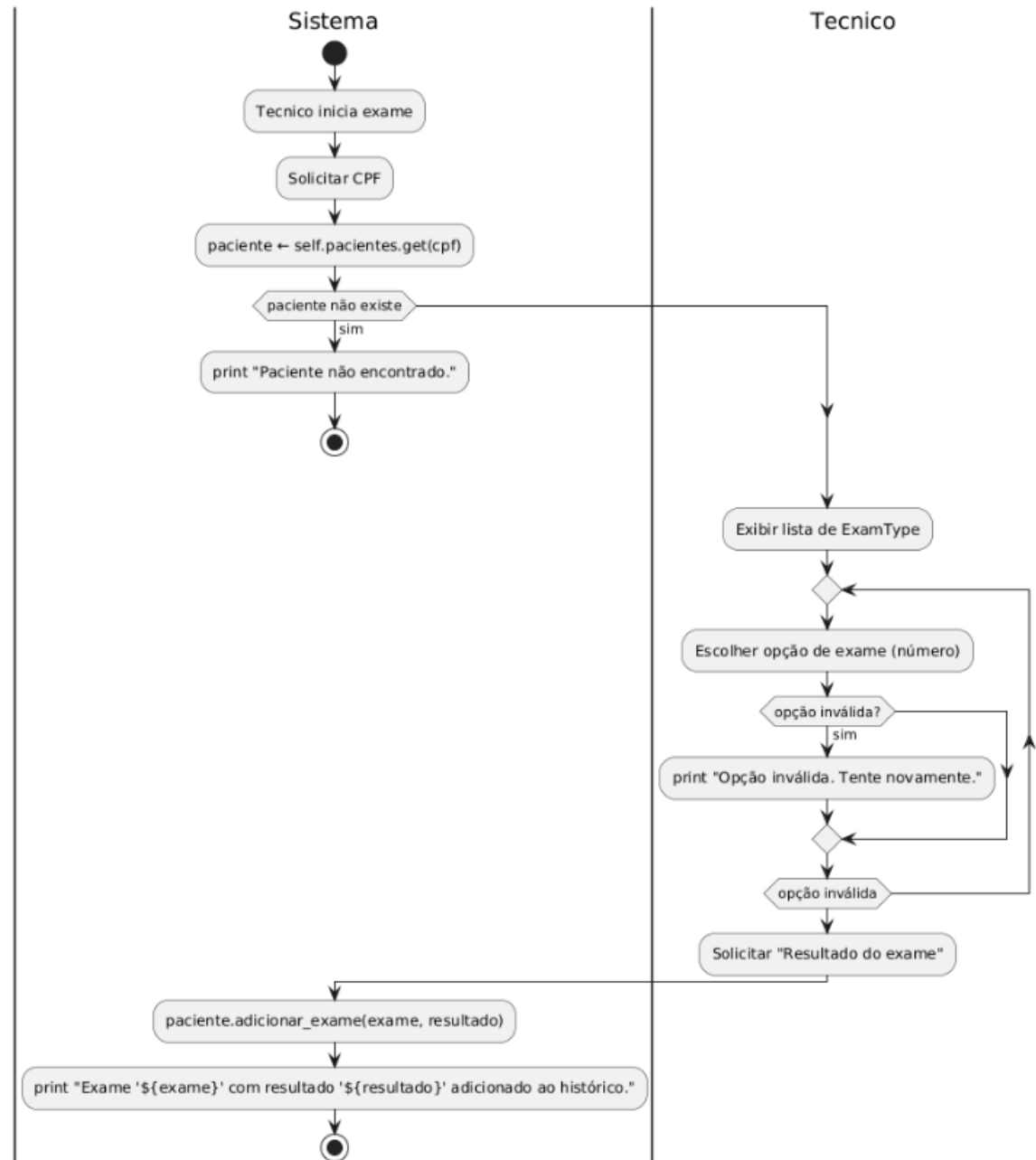


# Consulta





# Exame



# Herança

**Superclasse Profissional** reúne atributos e métodos comuns (nome, registro, login, hash de senha, contratos para cadastro/edição de pacientes) profissionais.

**Enfermeiro** herda autenticação e adiciona triagem(paciente: Paciente) para coletar sinais vitais e gerar Anamnese.

**Medico** herda e adiciona sugerir\_diagnostics(paciente: Paciente) além de geração de receituário e declaração (produz objetos Diagnostico) .

**Tecnico** herda e implementa adicionar\_exame\_sistema(paciente: Paciente) para registrar resultados de exames profissionais.

# Polimorfismo

Na classe **SistemaMedicclass** (sistema.py), o método genérico login() retorna sempre um Profissional.

```
user = sistema.login()
# Polimorfismo: trata o retorno como Profissional,
# mas cada subclasse implementa seu próprio comportamento.
if hasattr(user, 'triagem'):
    user.triagem(paciente)    # Enf. executa triagem
elif hasattr(user, 'sugerir_diagnosticos'):
    user.sugerir_diagnosticos(paciente) # Med. faz diagnóstico
```

# Abstração e encapsulamento

**Módulo Anamnese:** define a classe Anamnese, que encapsula tudo que é relevante à triagem (freq. cardíaca, pressão, oximetria, perguntas e respostas, tipo de sintoma, timestamp) e expõe apenas to\_dict() para uso externo anamnese.

**Classe Paciente:** mantém seu histórico médico em arquivo privado (\_historico\_file) e fornece métodos públicos como registrar\_entrada(), atualizar\_historico() e consultar\_historico() para manipulação.

**Classe Abstrata Profissional:** oculta sua senha real em \_senha\_hash e só permite autenticação via autenticar(), impedindo acesso direto aos atributos profissionais.