UNIVERSIDADE DE PERNAMBUCO (UPE)

ENGENHARIA DE SOFTWARE

CLAUDENIRA NASCIMENTO

FELIPE DA SILVA

PROJETO: SISTEMA PARA UNIDADE BÁSICA DE SAÚDE (USB)

SURUBIM

2025

Diagrama

O conteúdo gerado por IA pode estar incorreto.

Diagrama de Casos de Uso - Sistema USB

' Atores

actor "Agente de Saúde" as AS

actor "Enfermeiro" as ENF

' Limite do Sistema

rectangle "Sistema USB" {

usecase UC1 as "1. Autenticar Usuário"

usecase UC2 as "2. Gerenciar Famílias (CRUD)"

usecase UC3 as "3. Registrar Visita Domiciliar"

usecase UC4 as "4. Gerar e Salvar Relatório"

usecase UC5 as "5. Visualizar Relatórios"

' Casos de Uso Incluídos (Detalhes do CRUD)

usecase UC6 as "Registrar Família"

usecase UC7 as "Listar Famílias/Moradores"

usecase UC8 as "Editar Família/Morador"

usecase UC9 as "Remover Família/Morador"

}

' Relações de Inclusão (Detalhamento do Gerenciamento)

UC2 .> UC6 : <<include>>

UC2 .> UC7 : <<include>>

UC2 .> UC8 : <<include>>

UC2 .> UC9 : <<include>>

' Relações entre Atores e Casos de Uso

' Funcionalidades Comuns (Autenticação, Relatórios)

AS -- UC1

ENF -- UC1

AS -- UC4

ENF -- UC4

AS -- UC5

ENF -- UC5

' Funcionalidades do Agente de Saúde (Gestão de Famílias e Visitas)

AS -- UC2

AS -- UC3

Diagrama

O conteúdo gerado por IA pode estar incorreto.

Diagrama de Classes - Sistema USB

' 1. CLASSES ABSTRATAS E PRINCIPAIS

abstract class Pessoa {

# cpf : String

# nome : String

--

+ getCpf() : String

+ getNome() : String

}

class Morador {

- sus : String

- dataNascimento : String

- idade : int

- sexo : String

- condicoesSaude : String

- riscoSaude : boolean

--

+ toPersistenciaString() : String

}

class AgenteDeSaude {

- areaAtuacao : String

--

+ {static} familias : ArrayList<Familia>

+ {static} autenticar(Scanner) : boolean

+ {static} executar(Scanner) : void

+ {static} registrar(Scanner) : void

' (Demais métodos CRUD static)

}

class Enfermeiro {

- especialidade : String

--

+ {static} relatorios : List<Relatorio>

+ {static} autenticar(Scanner) : boolean

+ {static} executar(Scanner) : void

' (Demais métodos static)

}

class Familia {

- responsavel : Morador

- endereco : String

- moradores : List<Morador>

- visitas : List<VisitaDomiciliar>

--

+ adicionarMorador(Morador) : void

+ toPersistenciaString() : String

' (Demais métodos CRUD)

}

class VisitaDomiciliar {

- data : Date

- observacoes : String

- agente : AgenteDeSaude

- familiaVisitada : Familia

--

+ salvarEmArquivo() : void

+ toPersistenciaString() : String

}

class Relatorio {

+ {static} relatorios : List<Relatorio>

- dataGeracao : Date

- tipo : String

- conteudo : String

--

+ {static} executar(Scanner) : void

+ salvarEmArquivo() : void

}

class ArquivoUtil << Utility >> {

+ {static} salvar(String, String) : void

+ {static} sobrescrever(String, String) : void

+ {static} ler(String) : List<String>

+ {static} carregarFamilias() : ArrayList<Familia>

}

' 2. HERANÇA

Pessoa <|-- Morador

Pessoa <|-- AgenteDeSaude

Pessoa <|-- Enfermeiro

' 3. ASSOCIAÇÕES (Atributos de Lista e Referência)

' 3.1. Relações da Família

' A Família é composta por muitos Moradores (Lista)

Familia "1" -- "\*" Morador : moradores

' A Família tem exatamente um Morador como responsável

Familia "1" -- "1" Morador : responsável

' A Família registra muitas Visitas

Familia "1" -- "\*" VisitaDomiciliar : visitas

' 3.2. Relações da Visita Domiciliar

' A Visita foi feita por um Agente

VisitaDomiciliar "1" -- "1" AgenteDeSaude : agente

' A Visita está ligada a uma Família

VisitaDomiciliar "1" -- "1" Familia : familiaVisitada

' 3.3. Relação Morador - Visita (Para Moradores)

' Não explícita, mas Visita Domiciliar usa Morador indiretamente via Família.