Documentação de Projeto

para o sistema

ConectaAlimento

Versão 1.0

Projeto de sistema elaborado pelo(s) aluno(s) Isabella Luiza Dias dos Santos
como parte da disciplina **Projeto de Software**.

26 de Outubro de 2025

Tabela de Conteúdo

Histórico de Revisões

Nome	Data	Razões para Mudança	Versão
Isabella Luiza D. S.	10/11	Alterações Necessárias nos Diagramas de Sequência (Subseção 3.4)	0.1
Isabella Luiza D. S.	15/11	Alterações nos Diagramas de Comunicação (Subseção 3.5)	0.2
Isabella Luiza D. S.	16/11	Alterações no Diagrama DER (Subseção 3.6)	0.3

1. Introdução

Este documento agrega:

1) A elaboração e revisão de modelos de domínio.

2) Modelos de projeto para o sistema ConectaAlimento.

A referência principal para a descrição geral do problema, domínio e requisitos do sistema é o documento de especificação que descreve a visão de domínio do sistema.

2. Modelos de Usuário e Requisitos

2.1 Descrição de Atores

Nesta subseção são apresentados os atores que participam do sistema, juntamente com suas funções e responsabilidades no processo de doação, transporte e distribuição de alimentos.

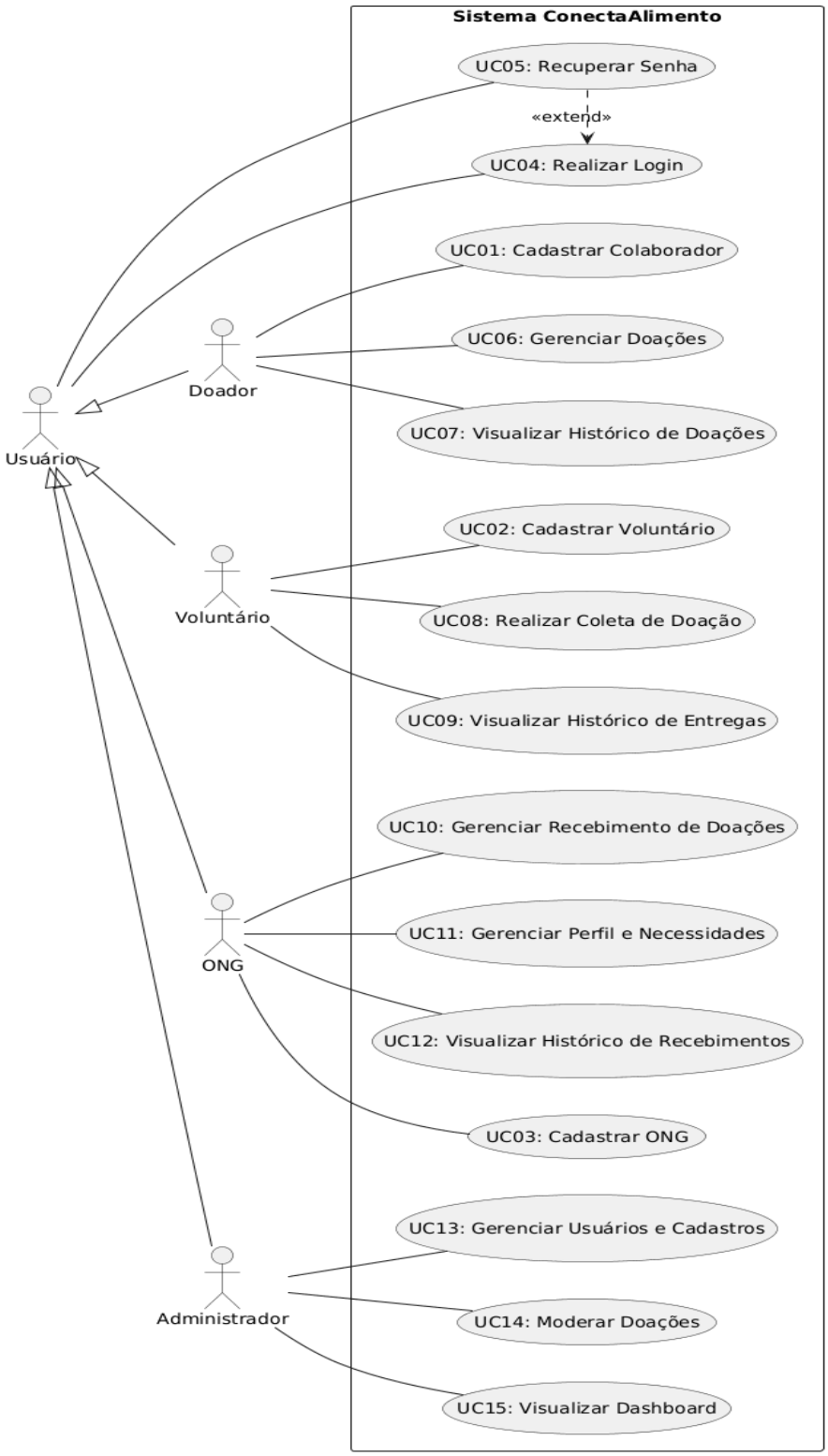
Doador: Entidade, seja pessoa jurídica ou pessoa física, responsável por disponibilizar alimentos excedentes para doação.

Voluntário: Pessoa física devidamente habilitada para realizar o transporte dos alimentos, garantindo a entrega do Doador até a Instituição receptora.

Instituição: Organizações sem fins lucrativos (ONGs) que recebem os alimentos doados e os destinam à população em situação de vulnerabilidade social.

Administrador: Agente responsável pela validação dos cadastros dos usuários, bem como pela manutenção e gestão do sistema.

2.2 Modelo de Casos de Uso



Código - PlantUML:

```
@startuml
left to right direction

actor Usuário
actor Doador
actor Voluntário
actor ONG
actor Administrador as Admin

Usuário <|-- Doador
Usuário <|-- Voluntário
Usuário <|-- ONG
Usuário <|-- Admin

rectangle "Sistema ConectaAlimento" {

  usecase "UC01: Cadastrar Colaborador" as UC01
  usecase "UC02: Cadastrar Voluntário" as UC02
  usecase "UC03: Cadastrar ONG" as UC03
  usecase "UC04: Realizar Login" as UC04
  usecase "UC05: Recuperar Senha" as UC05

  usecase "UC06: Gerenciar Doações" as UC06
  usecase "UC07: Visualizar Histórico de Doações" as UC07

  usecase "UC08: Realizar Coleta de Doação" as UC08
  usecase "UC09: Visualizar Histórico de Entregas" as UC09

  usecase "UC10: Gerenciar Recebimento de Doações" as UC10
  usecase "UC11: Gerenciar Perfil e Necessidades" as UC11
  usecase "UC12: Visualizar Histórico de Recebimentos" as UC12

  usecase "UC13: Gerenciar Usuários e Cadastros" as UC13
  usecase "UC14: Moderar Doações" as UC14
  usecase "UC15: Visualizar Dashboard" as UC15

  UC05 .> UC04 : <<extend>>
}

Usuário -- UC04
Usuário -- UC05
```

Doador -- UC01
Voluntário -- UC02
ONG -- UC03

Doador -- UC06
Doador -- UC07

Voluntário -- UC08
Voluntário -- UC09

ONG -- UC10
ONG -- UC11
ONG -- UC12

Admin -- UC13
Admin -- UC14
Admin -- UC15
@enduml

Histórias de Usuário (User Stories):

UC01 - Cadastrar Colaborador

Como um colaborador (restaurante ou evento), Eu desejo me cadastrar na plataforma, Para poder registrar sobras de alimentos para doação.

UC02 - Cadastrar Voluntário

Como um voluntário, Eu desejo me cadastrar no sistema informando meus dados (além da CNH), Para poder aceitar e realizar o transporte de doações.

UC03 - Cadastrar ONG

Como uma ONG, Eu desejo me cadastrar no sistema (enviando CNPJ e documentos), Para ser validada e listada como beneficiária para receber doações.

UC04 - Realizar Login

Como um usuário (Doador, Voluntário, ONG ou Admin), Eu desejo realizar login com meu e-mail e senha, Para acessar as funcionalidades específicas da minha conta.

UC05 - Recuperar Senha

Como um usuário, Eu desejo solicitar a recuperação da minha senha, Para poder acessar o sistema caso eu a esqueça.

UC06 - Gerenciar Doações

Como um Doador, Eu desejo cadastrar, acompanhar o status e confirmar a retirada das minhas doações, Para garantir que o alimento chegue ao destino.

UC07 - Visualizar Histórico de Doações

Como um Doador, Eu desejo visualizar meu histórico de doações, Para ter um registro do impacto social que gerei e para minha própria organização.

UC08 - Realizar Coleta de Doação

Como um Voluntário, Eu desejo visualizar, aceitar e registrar os passos (retirada e entrega) de uma coleta, Para transportar o alimento do Doador até a ONG.

UC09 - Visualizar Histórico de Entregas

Como um Voluntário, Eu desejo ver meu histórico de entregas realizadas, Para acompanhar minhas atividades de voluntariado.

UC10 - Gerenciar Recebimento de Doações

Como uma ONG, Eu desejo visualizar doações a caminho e confirmar o recebimento delas, Para gerenciar meu estoque e validar a entrega.

UC11 - Gerenciar Perfil e Necessidades

Como uma ONG, Eu desejo atualizar meu perfil e listar minhas necessidades (tipos de alimentos), Para que o sistema possa me alocar as doações mais adequadas.

UC12 - Visualizar Histórico de Recebimentos

Como uma ONG, Eu desejo consultar o histórico de alimentos que recebi, Para auxiliar na minha prestação de contas, gestão interna e relatórios de impacto.

UC13 - Gerenciar Usuários e Cadastros

Como um Administrador, Eu desejo validar novos cadastros (ONGs, Voluntários) e gerenciar usuários existentes (bloquear/desativar), Para manter a segurança e a integridade da plataforma.

UC14 - Moderar Doações

Como um Administrador, Eu desejo monitorar os pedidos de doação cadastrados, Para remover publicações inadequadas.

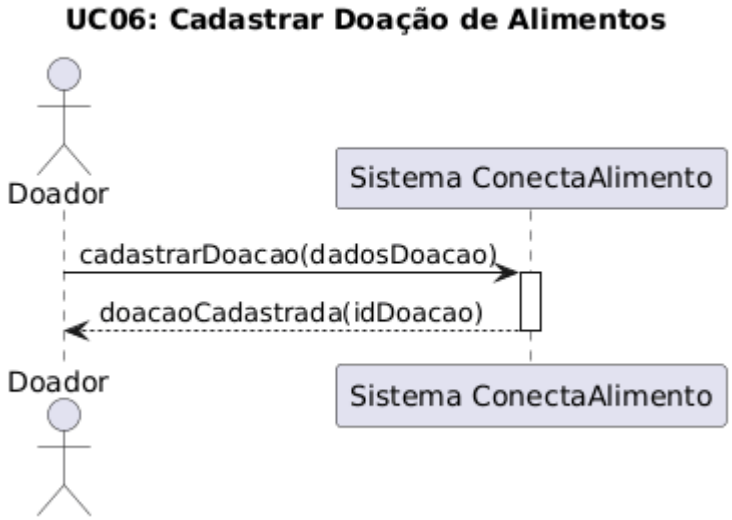
UC15 - Visualizar Dashboard do Sistema

Como um Administrador, Eu desejo visualizar um dashboard com estatísticas gerais, Para monitorar o impacto e o funcionamento do sistema.

2.3 Diagrama de Sequência do Sistema

Nesta subseção é apresentado o diagrama de sequência do sistema de pelo menos, 3 Casos de Uso ou Histórias de Usuário descritos na Seção 2.2.

UC06: Gerenciar Doações (Cadastrar Doação)



Código PlantUML:

@startuml
title UC06: Cadastrar Doação de Alimentos

actor Doador
participant "Sistema ConectaAlimento" as Sistema

Doador -> Sistema : cadastrarDoacao(dadosDoacao)

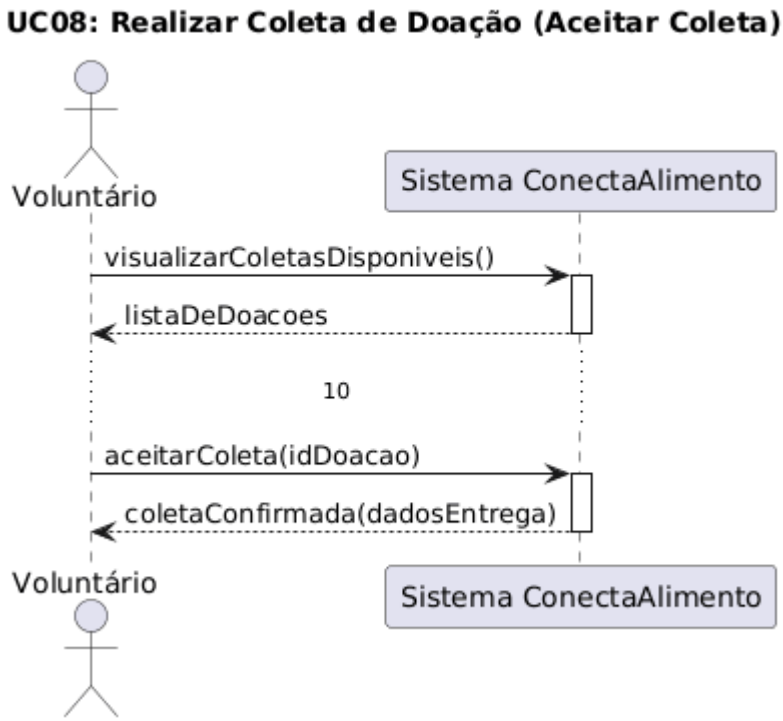
activate Sistema

Sistema --> Doador : doacaoCadastrada(idDoacao)

deactivate Sistema
@enduml

Contrato	
Operação	cadastrarDoacao(dadosDoacao: DtoDoacao)
Referências cruzadas	UC06: Gerenciar Doações
Pré-condições	1. O Doador deve estar autenticado no sistema.
Pós-condições	1. Uma nova instância de Doacao foi criada no sistema. 2. A Doacao foi associada ao Doador que a cadastrou. 3. O estado da Doacao foi definido como "Aguardando Voluntário". 4. Um log da transação foi criado.

UC08: Realizar Coleta de Doação (Aceitar Coleta)



Código PlantUML:

@startuml
title UC08: Realizar Coleta de Doação (Aceitar Coleta)

actor Voluntário

participant "Sistema ConectaAlimento" as Sistema

Voluntário -> Sistema : visualizarColetasDisponiveis()
activate Sistema

Sistema --> Voluntário : listaDeDoacoes
deactivate Sistema

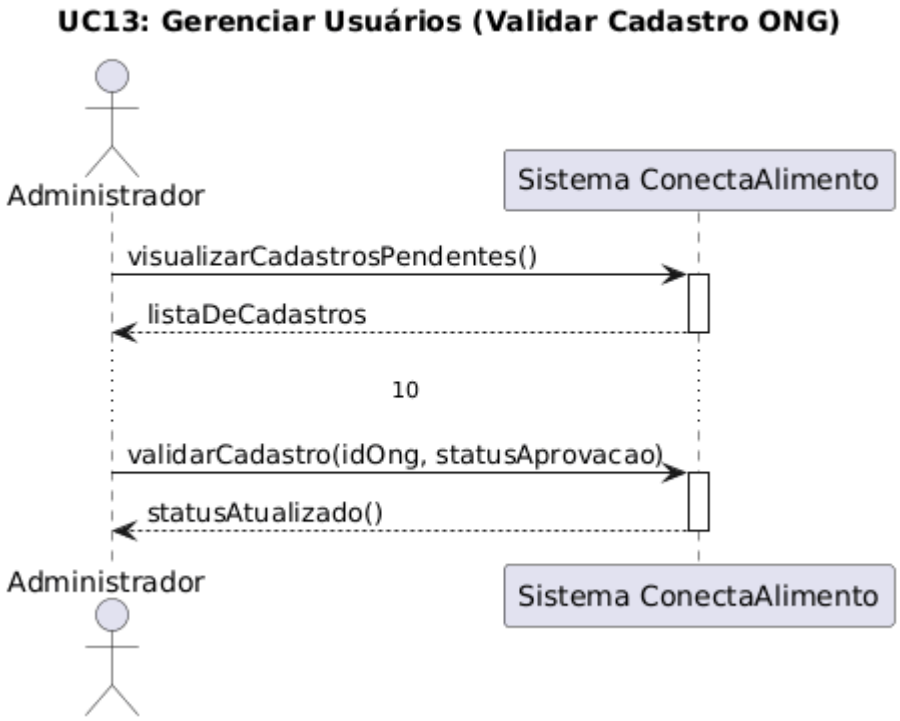
... 10 ...

Voluntário -> Sistema : aceitarColeta(idDoacao)
activate Sistema

Sistema --> Voluntário : coletaConfirmada(dadosEntrega)
deactivate Sistema
@enduml

Contrato	
Operação	aceitarColeta(idDoacao: Inteiro)
Referências cruzadas	UC08: Realizar Coleta de Doação
Pré-condições	1. O Voluntário deve estar autenticado no sistema 2. A Doacao referenciada por idDoacao deve existir 3. O estado da Doacao deve ser "Aguardando Voluntário"
Pós-condições	1. O estado da Doacao foi alterado para "Coleta Aceita" 2. A Doacao foi associada ao Voluntário 3.O Doador e a ONG de destino foram notificados.

UC13: Gerenciar Usuários e Cadastros (Validar ONG)



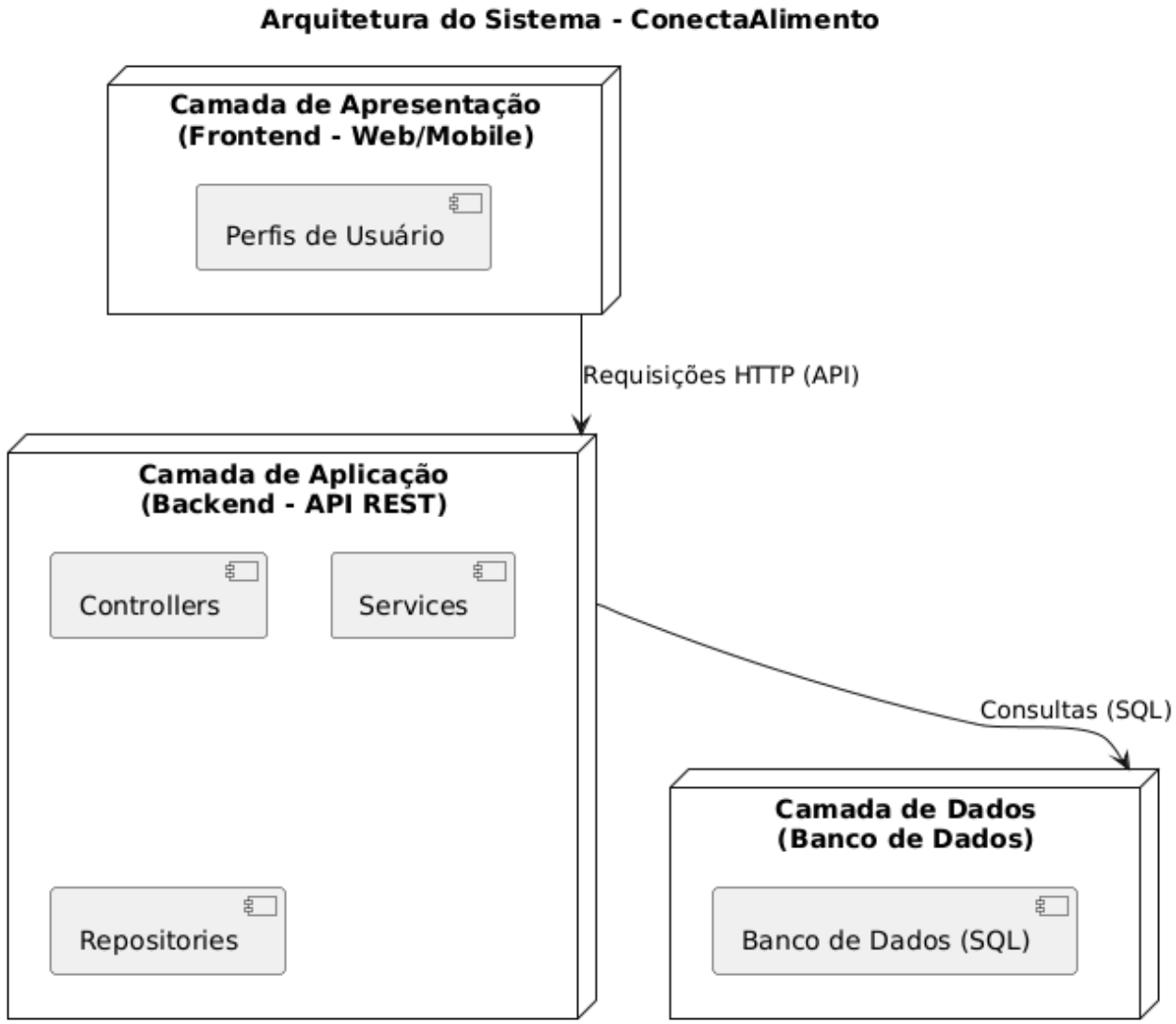
Código PlantUML:

```
@startuml
title UC13: Gerenciar Usuários (Validar Cadastro ONG)

actor Administrador as Admin
participant "Sistema ConectaAlimento" as Sistema

Admin -> Sistema : visualizarCadastrosPendentes()
activate Sistema

Sistema --> Admin : listaDeCadastros
deactivate Sistema

... 10 ...

Admin -> Sistema : validarCadastro(idOng, statusAprovacao)
activate Sistema

Sistema --> Admin : statusAtualizado()
deactivate Sistema
@enduml
```

Contrato	
Operação	validarCadastro(idUsuario: Inteiro, statusAprovacao: Booleano)
Referências cruzadas	UC13: Gerenciar Usuários e Cadastros
Pré-condições	1. O Administrador deve estar autenticado no sistema 2. O Usuario (ONG) referido por idUsuario deve existir. 3. O estado do Usuario (ONG) deve ser "Aguardando Validação"
Pós-condições	1. O estado do Usuario (ONG) foi alterado para "Ativo" ou "Rejeitado". 2. A ONG foi notificada sobre a decisão.

3. Modelos de Projeto

3.1 Arquitetura



Código PlantUML:

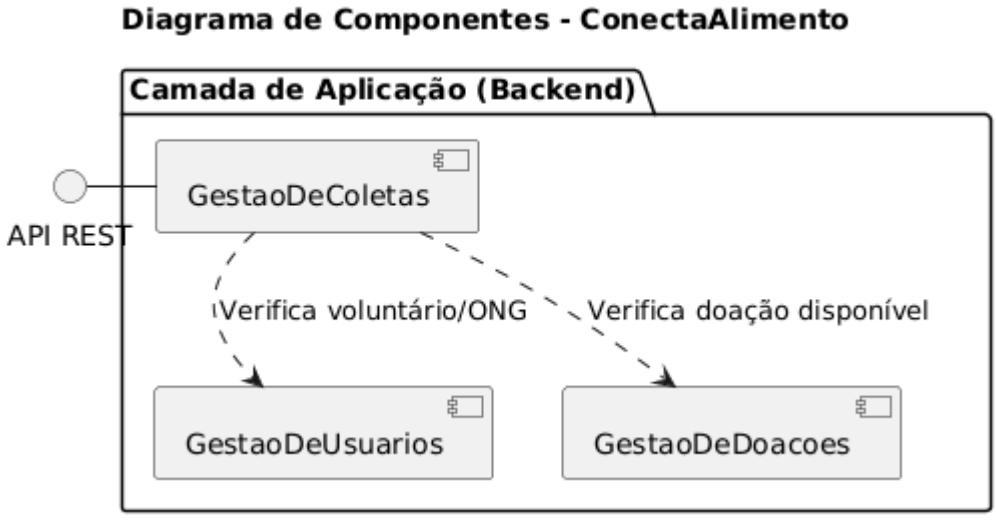
```
@startuml
title Arquitetura do Sistema - ConectaAlimento

top to bottom direction

node "Camada de Apresentação\n(Frontend - Web/Mobile)" as Frontend {
  component [Perfis de Usuário]
}
```

```
node "Camada de Aplicação\n(Backend - API REST)" as Backend {
   component [Controllers]
   component [Services]
   component [Repositories]
}

node "Camada de Dados\n(Banco de Dados)" as DB {
   component [Banco de Dados (SQL)]
}

Frontend --> Backend : "Requisições HTTP (API)"
Backend --> DB : "Consultas (SQL)"
@enduml
```

3.2 Diagrama de Componentes e Implantação.

Diagramas de componentes do sistema. Diagrama de implantação mostrando onde os componentes estarão alocados para a execução.

Diagrama de Componentes:

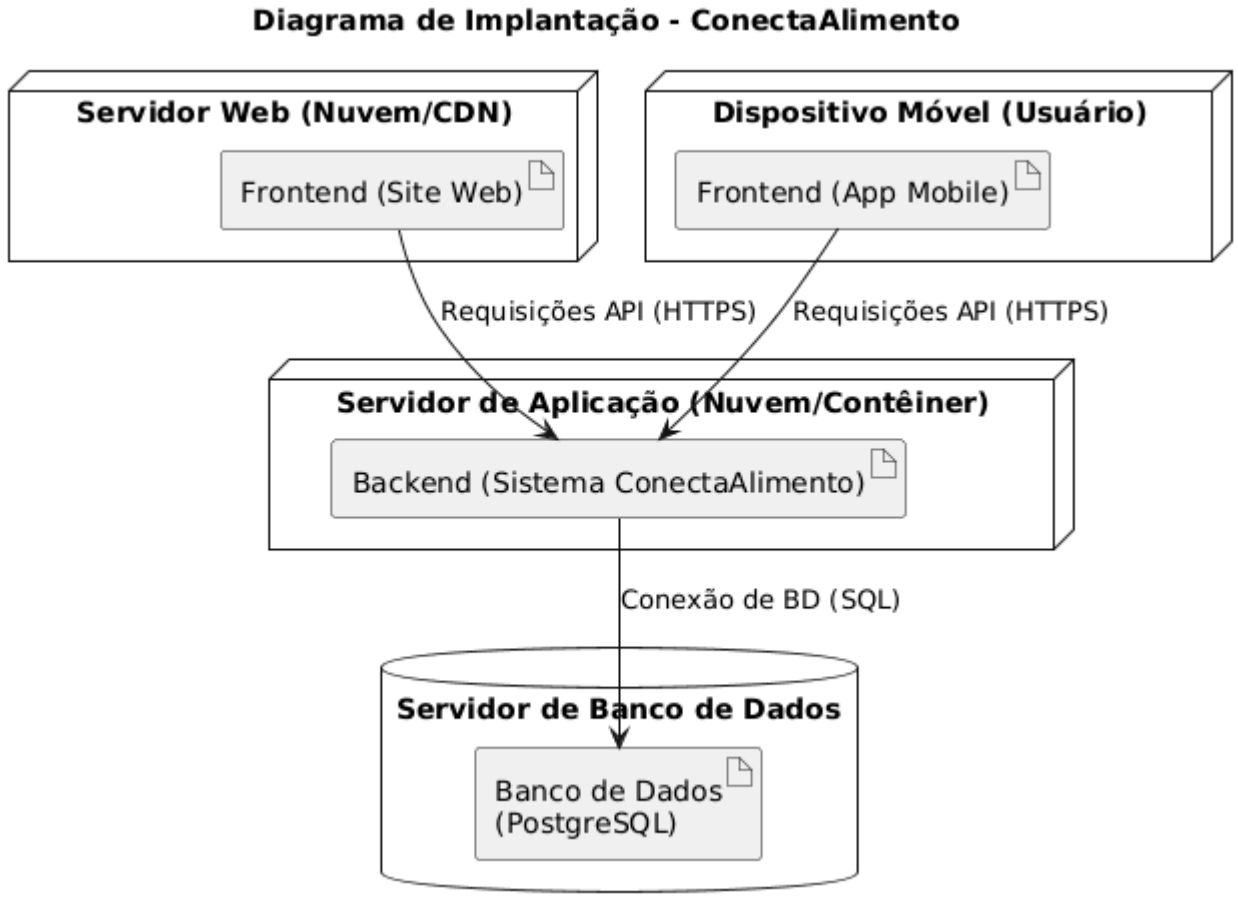


Código PlantUML:

```
@startuml
title Diagrama de Componentes - ConectaAlimento
interface "API REST" as API

package "Camada de Aplicação (Backend)" {
```

```
   component [GestaoDeColetas] as Coletas
   component [GestaoDeUsuarios] as Usuarios
   component [GestaoDeDoacoes] as Doacoes

}

API - [Coletas]

[Coletas] ..> [Usuarios] : "Verifica voluntário/ONG"
[Coletas] ..> [Doacoes] : "Verifica doação disponível"
@enduml
```

Diagrama de Implantação:

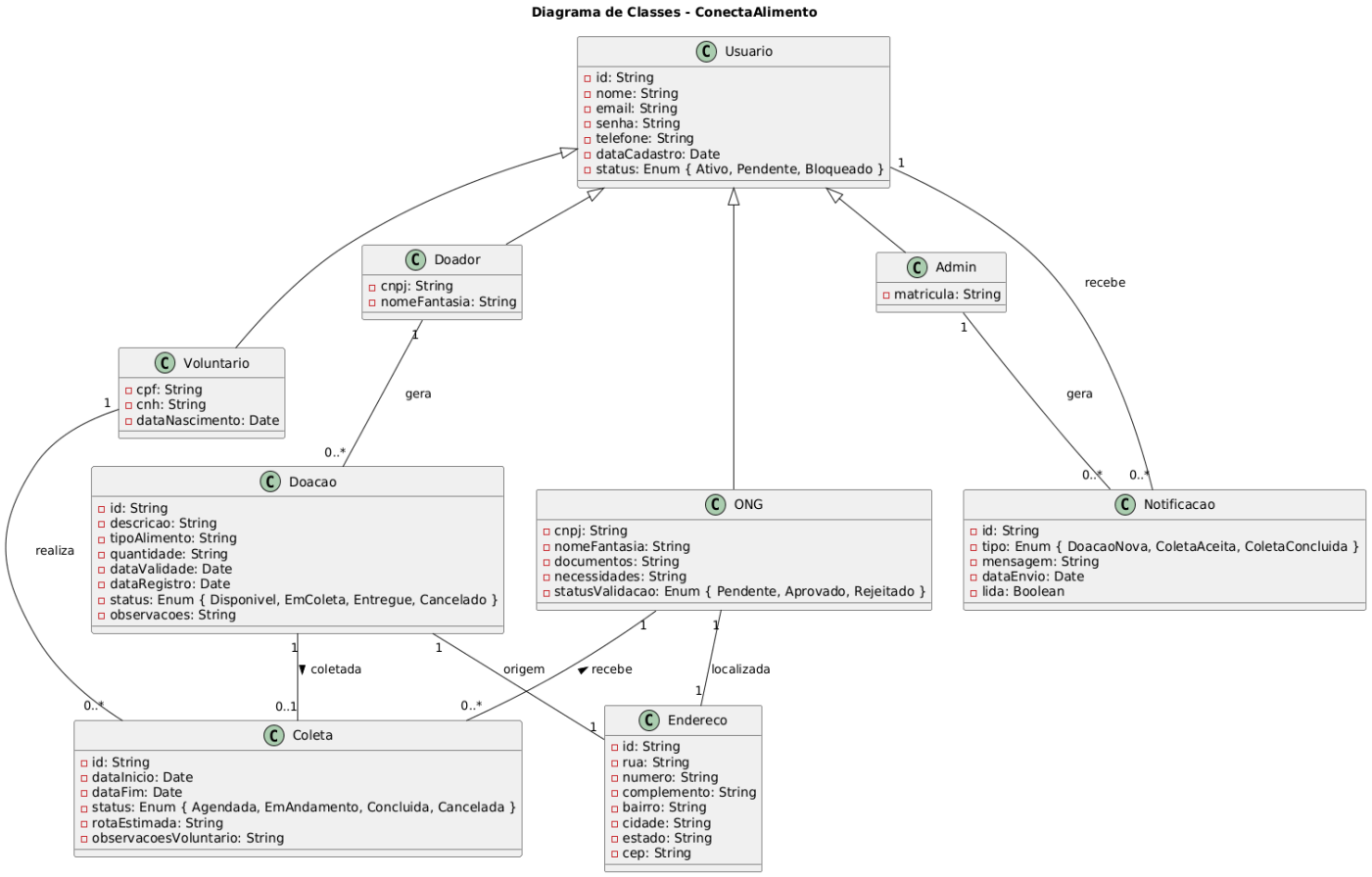


Código PlantUML:

```
@startuml
title Diagrama de Implantação - ConectaAlimento

database "Servidor de Banco de Dados" as DB {
   artifact "Banco de Dados\n(PostgreSQL)" as BD
}

node "Servidor de Aplicação (Nuvem/Contêiner)" as AppServer {
   artifact "Backend (Sistema ConectaAlimento)" as Backend
}

node "Servidor Web (Nuvem/CDN)" as WebServer {
   artifact "Frontend (Site Web)" as WebApp
}

node "Dispositivo Móvel (Usuário)" as Mobile {
   artifact "Frontend (App Mobile)" as MobileApp
}

WebApp --> Backend : "Requisições API (HTTPS)"
MobileApp --> Backend : "Requisições API (HTTPS)"
Backend --> BD : "Conexão de BD (SQL)"
@enduml
```

3.3 Diagrama de Classes

Diagrama de classes do sistema.



Código PlantUML:

@startuml
title Diagrama de Classes - ConectaAlimento

class Usuario {
  - id: String
  - nome: String
  - email: String
  - senha: String
  - telefone: String
  - dataCadastro: Date
  - status: Enum { Ativo, Pendente, Bloqueado }
}

class Doador {

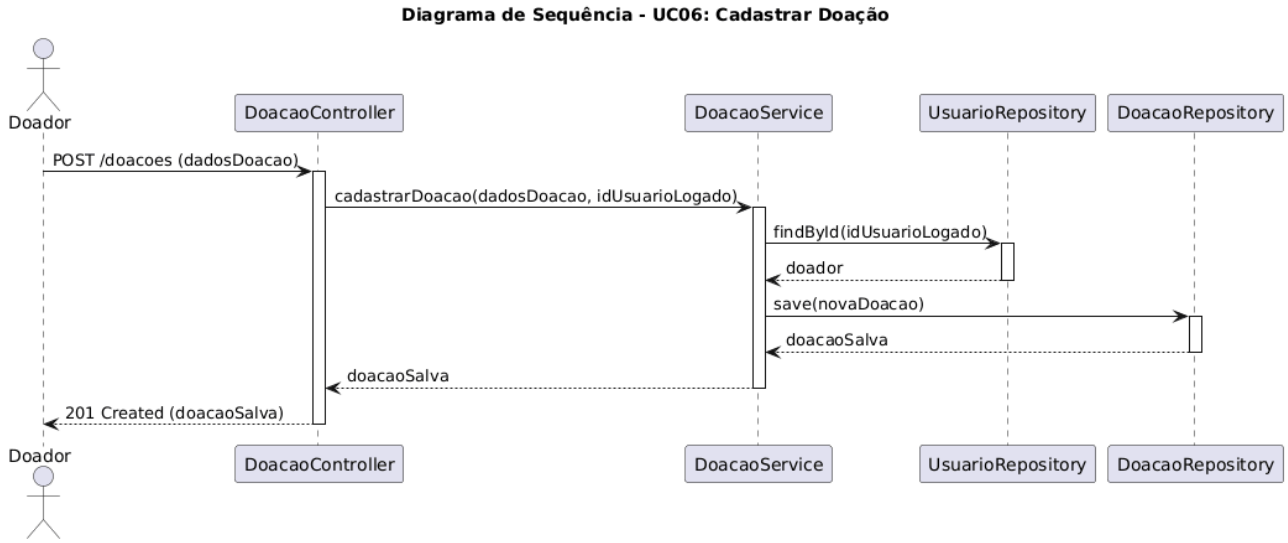
```
 - cnpj: String
 - nomeFantasia: String
}

class Voluntario {
 - cpf: String
 - cnh: String
 - dataNascimento: Date
}

class ONG {
 - cnpj: String
 - nomeFantasia: String
 - documentos: String
 - necessidades: String
 - statusValidacao: Enum { Pendente, Aprovado, Rejeitado }
}

class Admin {
 - matricula: String
}

Usuario <|-- Doador
Usuario <|-- Voluntario
Usuario <|-- ONG
Usuario <|-- Admin

class Doacao {
 - id: String
 - descricao: String
 - tipoAlimento: String
 - quantidade: String
 - dataValidade: Date
 - dataRegistro: Date
 - status: Enum { Disponivel, EmColeta, Entregue, Cancelado }
 - observacoes: String
}

class Coleta {
 - id: String
 - dataInicio: Date
 - dataFim: Date
 - status: Enum { Agendada, EmAndamento, Concluida, Cancelada }
 - rotaEstimada: String
```

```
  - observacoesVoluntario: String
}

class Endereco {
  - id: String
  - rua: String
  - numero: String
  - complemento: String
  - bairro: String
  - cidade: String
  - estado: String
  - cep: String
}

class Notificacao {
  - id: String
  - tipo: Enum { DoacaoNova, ColetaAceita, ColetaConcluida }
  - mensagem: String
  - dataEnvio: Date
  - lida: Boolean
}


Doador "1" -- "0..*" Doacao : gera

Doacao "1" -- "0..1" Coleta : coletada >

Voluntario "1" -- "0..*" Coleta : realiza

ONG "1" -- "0..*" Coleta : recebe <

Doacao "1" -- "1" Endereco : origem

ONG "1" -- "1" Endereco : localizada

Usuario "1" -- "0..*" Notificacao : recebe

Admin "1" -- "0..*" Notificacao : gera
@enduml
```

3.4 Diagramas de Sequência

Diagramas de sequência para realização de casos de uso.

Diagrama de Sequência - UC06: Cadastrar Doação



Código PlantUML:

@startuml
title Diagrama de Sequência - UC06: Cadastrar Doação

actor Doador
participant "DoacaoController" as Controller
participant "DoacaoService" as Service
participant "UsuarioRepository" as UserRepo
participant "DoacaoRepository" as DoacaoRepo

Doador -> Controller : POST /doacoes (dadosDoacao)
activate Controller

Controller -> Service : cadastrarDoacao(dadosDoacao, idUsuarioLogado)
activate Service

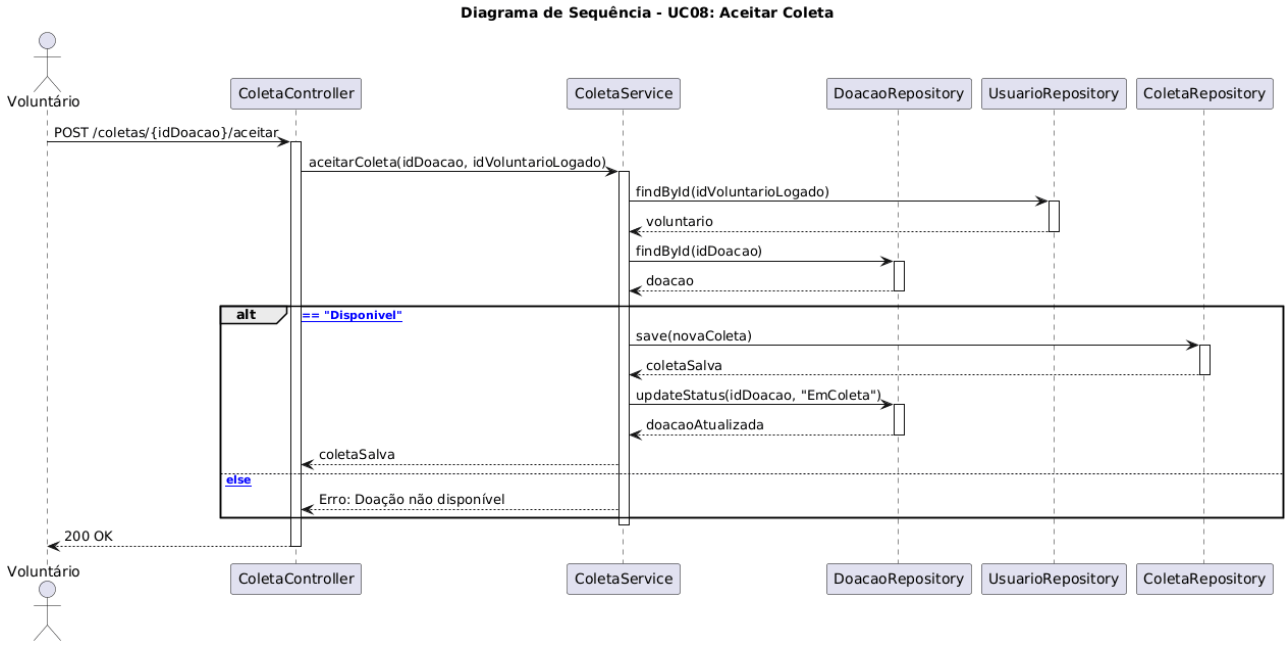
Service -> UserRepo : findById(idUsuarioLogado)
activate UserRepo
UserRepo --> Service : doador
deactivate UserRepo

Service -> DoacaoRepo : save(novaDoacao)
activate DoacaoRepo
DoacaoRepo --> Service : doacaoSalva
deactivate DoacaoRepo

Service --> Controller : doacaoSalva
deactivate Service

Controller --> Doador : 201 Created (doacaoSalva)
deactivate Controller
@enduml

Diagrama de Sequência - UC08: Aceitar Coleta



Código PlantUML:

@startuml
title Diagrama de Sequência - UC08: Aceitar Coleta

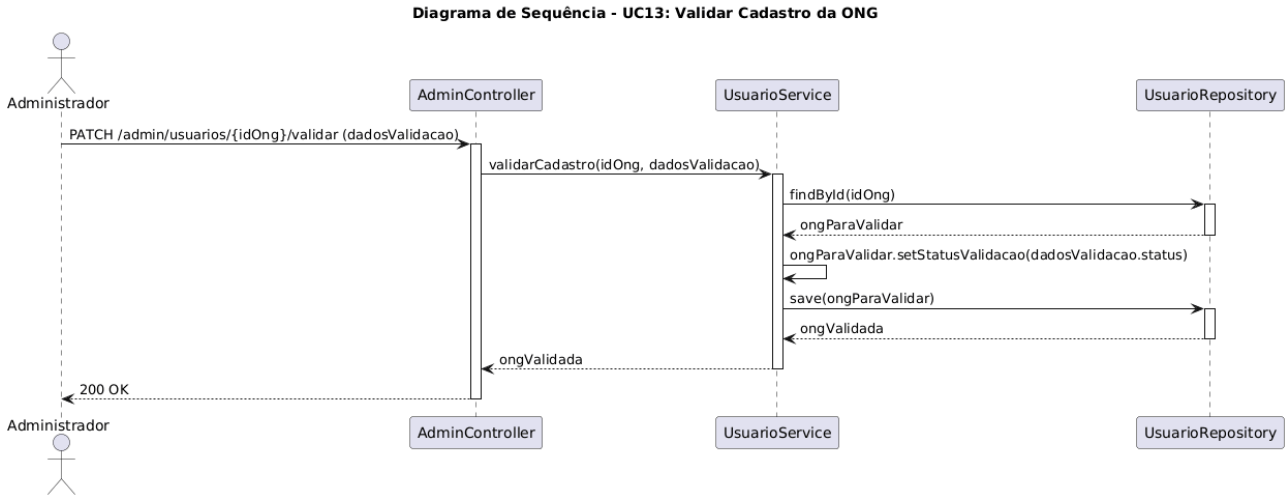
actor Voluntário
participant "ColetaController" as Controller
participant "ColetaService" as Service
participant "DoacaoRepository" as DoacaoRepo
participant "UsuarioRepository" as UserRepo
participant "ColetaRepository" as ColetaRepo

Voluntário -> Controller : POST /coletas/{idDoacao}/aceitar
activate Controller

Controller -> Service : aceitarColeta(idDoacao, idVoluntarioLogado)
activate Service

```
Service -> UserRepo : findById(idVoluntarioLogado)
activate UserRepo
UserRepo --> Service : voluntario
deactivate UserRepo

Service -> DoacaoRepo : findById(idDoacao)
activate DoacaoRepo
DoacaoRepo --> Service : doacao
deactivate DoacaoRepo

alt [doacao.status == "Disponivel"]
   Service -> ColetaRepo : save(novaColeta)
   activate ColetaRepo
   ColetaRepo --> Service : coletaSalva
   deactivate ColetaRepo

   Service -> DoacaoRepo : updateStatus(idDoacao, "EmColeta")
   activate DoacaoRepo
   DoacaoRepo --> Service : doacaoAtualizada
   deactivate DoacaoRepo

   Service --> Controller : coletaSalva
else [else]
   Service --> Controller : Erro: Doação não disponível
end

deactivate Service
Controller --> Voluntário : 200 OK
deactivate Controller
@enduml
```

Diagrama de Sequência - UC13: Validar Cadastro da ONG



Código PlantUML:

```
@startuml
title Diagrama de Sequência - UC13: Validar Cadastro da ONG

actor Administrador as Admin
participant "AdminController" as Controller
participant "UsuarioService" as Service
participant "UsuarioRepository" as UserRepo

Admin -> Controller : PATCH /admin/usuarios/{idOng}/validar (dadosValidacao)
activate Controller

Controller -> Service : validarCadastro(idOng, dadosValidacao)
activate Service

Service -> UserRepo : findById(idOng)
activate UserRepo
UserRepo --> Service : ongParaValidar
deactivate UserRepo

Service -> Service : ongParaValidar.setStatusValidacao(dadosValidacao.status)

Service -> UserRepo : save(ongParaValidar)
activate UserRepo
UserRepo --> Service : ongValidada
deactivate UserRepo
```

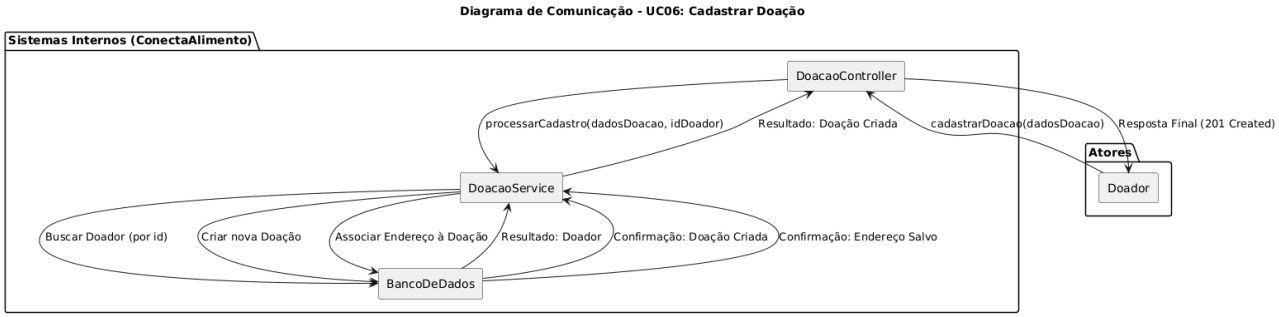
Service --> Controller : ongValidada
deactivate Service

Controller --> Admin : 200 OK
deactivate Controller
@enduml

3.5 Diagramas de Comunicação

Diagramas de comunicação para realização de casos de uso.

Diagrama de Comunicação - UC06: Cadastrar Doação



Código PlantUML:

@startuml
title Diagrama de Comunicação - UC06: Cadastrar Doação

package "Atores" {
  rectangle Doador
}

package "Sistemas Internos (ConectaAlimento)" {

  rectangle "DoacaoController" as Controller
  rectangle "DoacaoService" as Service

  rectangle "BancoDeDados" as DB
}

Doador -> Controller : cadastrarDoacao(dadosDoacao)

Controller -> Service : processarCadastro(dadosDoacao, idDoador)

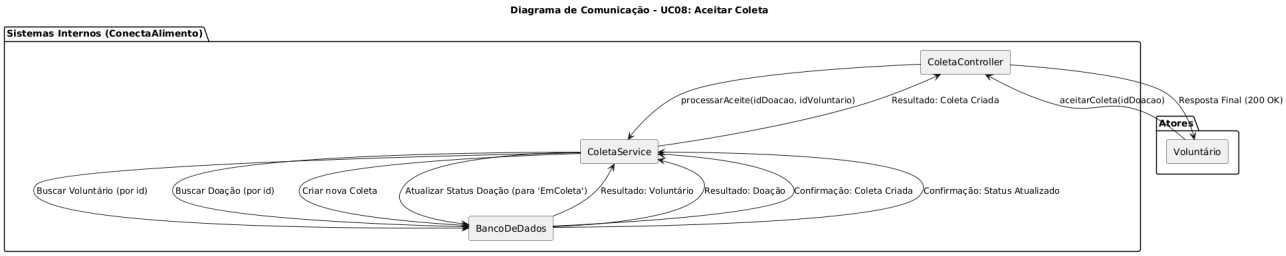
Service -> DB : "Buscar Doador (por id)"
DB --> Service : "Resultado: Doador"

Service -> DB : "Criar nova Doação"
DB --> Service : "Confirmação: Doação Criada"

Service -> DB : "Associar Endereço à Doação"
DB --> Service : "Confirmação: Endereço Salvo"

Service --> Controller : "Resultado: Doação Criada"
Controller --> Doador : "Resposta Final (201 Created)"
@enduml

Diagrama de Comunicação - UC08: Aceitar Coleta



Código PlantUML:

@startuml
title Diagrama de Comunicação - UC08: Aceitar Coleta

package "Atores" {
  rectangle Voluntário
}

package "Sistemas Internos (ConectaAlimento)" {

  rectangle "ColetaController" as Controller
  rectangle "ColetaService" as Service

  rectangle "BancoDeDados" as DB
}

Voluntário -> Controller : aceitarColeta(idDoacao)

Controller -> Service : processarAceite(idDoacao, idVoluntario)

Service -> DB : "Buscar Voluntário (por id)"

DB --> Service : "Resultado: Voluntário"

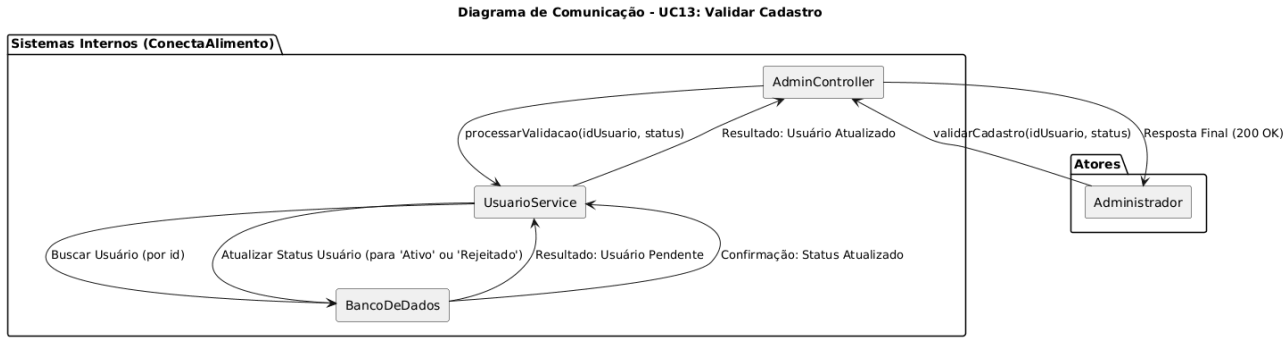
Service -> DB : "Buscar Doação (por id)"
DB --> Service : "Resultado: Doação"

Service -> DB : "Criar nova Coleta"
DB --> Service : "Confirmação: Coleta Criada"

Service -> DB : "Atualizar Status Doação (para 'EmColeta')"
DB --> Service : "Confirmação: Status Atualizado"

Service --> Controller : "Resultado: Coleta Criada"
Controller --> Voluntário : "Resposta Final (200 OK)"
@enduml

Diagrama de Comunicação - UC13: Validar Cadastro



Código PlantUML:

@startuml
title Diagrama de Comunicação - UC13: Validar Cadastro

package "Atores" {
 rectangle Administrador
}

package "Sistemas Internos (ConectaAlimento)" {

 rectangle "AdminController" as Controller
 rectangle "UsuarioService" as Service

 rectangle "BancoDeDados" as DB
}

Administrador -> Controller : validarCadastro(idUsuario, status)

Controller -> Service : processarValidacao(idUsuario, status)

Service -> DB : "Buscar Usuário (por id)"
DB --> Service : "Resultado: Usuário Pendente"

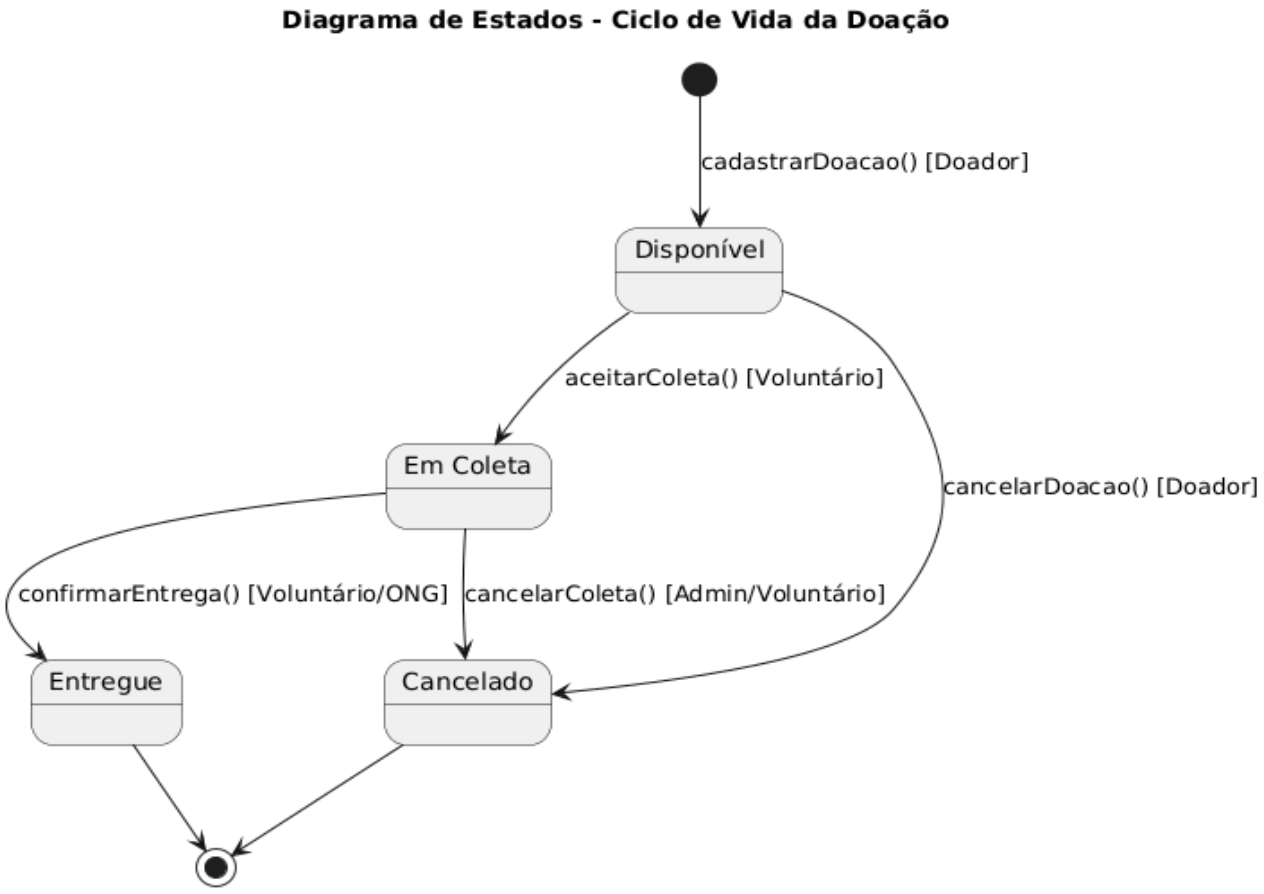
Service -> DB : "Atualizar Status Usuário (para 'Ativo' ou 'Rejeitado')"
DB --> Service : "Confirmação: Status Atualizado"

Service --> Controller : "Resultado: Usuário Atualizado"
Controller --> Administrador : "Resposta Final (200 OK)"
@enduml

3.6 Diagramas de Estados

Diagramas de estados do sistema.

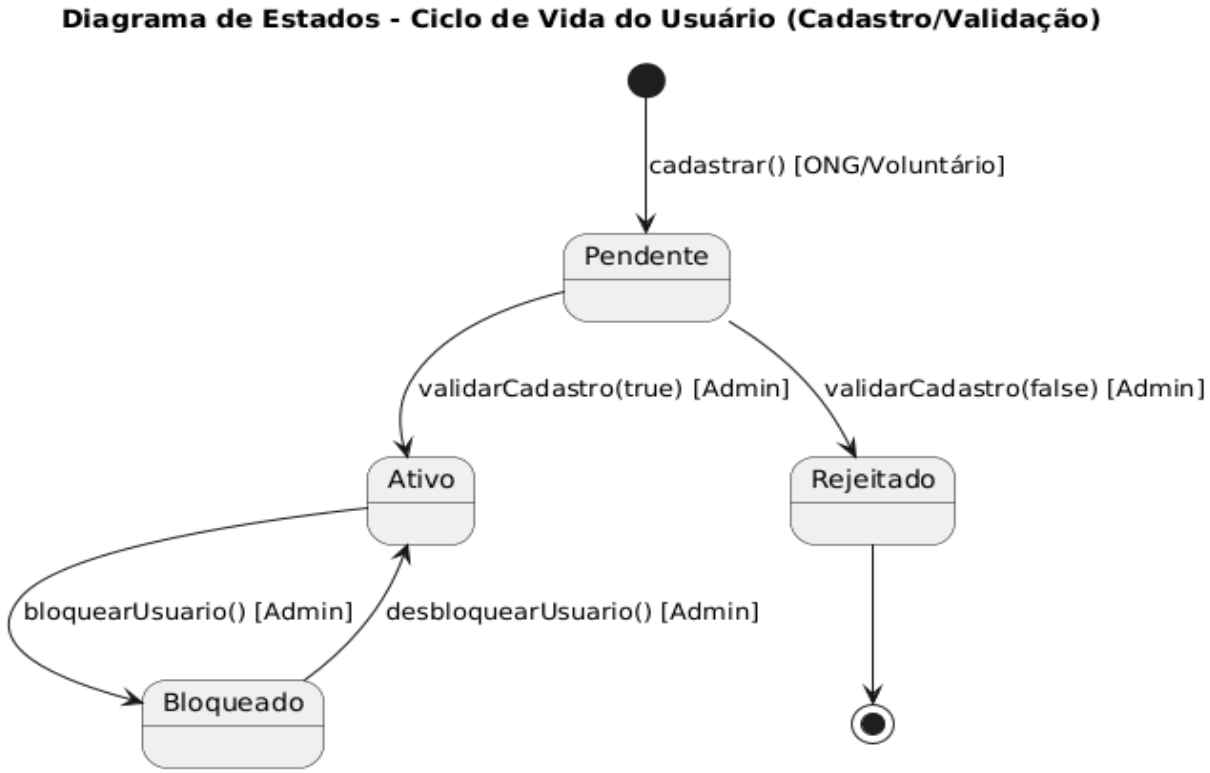
Diagrama de Estados - Ciclo de Vida da Doação



Código PlantUML:

```
@startuml
title Diagrama de Estados - Ciclo de Vida da Doação

state "Disponível" as Disponivel
state "Em Coleta" as EmColeta
state "Entregue" as Entregue
state "Cancelado" as Cancelado

[*] --> Disponivel : cadastrarDoacao() [Doador]

Disponivel --> EmColeta : aceitarColeta() [Voluntário]
EmColeta --> Entregue : confirmarEntrega() [Voluntário/ONG]

Disponivel --> Cancelado : cancelarDoacao() [Doador]
EmColeta --> Cancelado : cancelarColeta() [Admin/Voluntário]

Entregue --> [*]
Cancelado --> [*]
@enduml
```

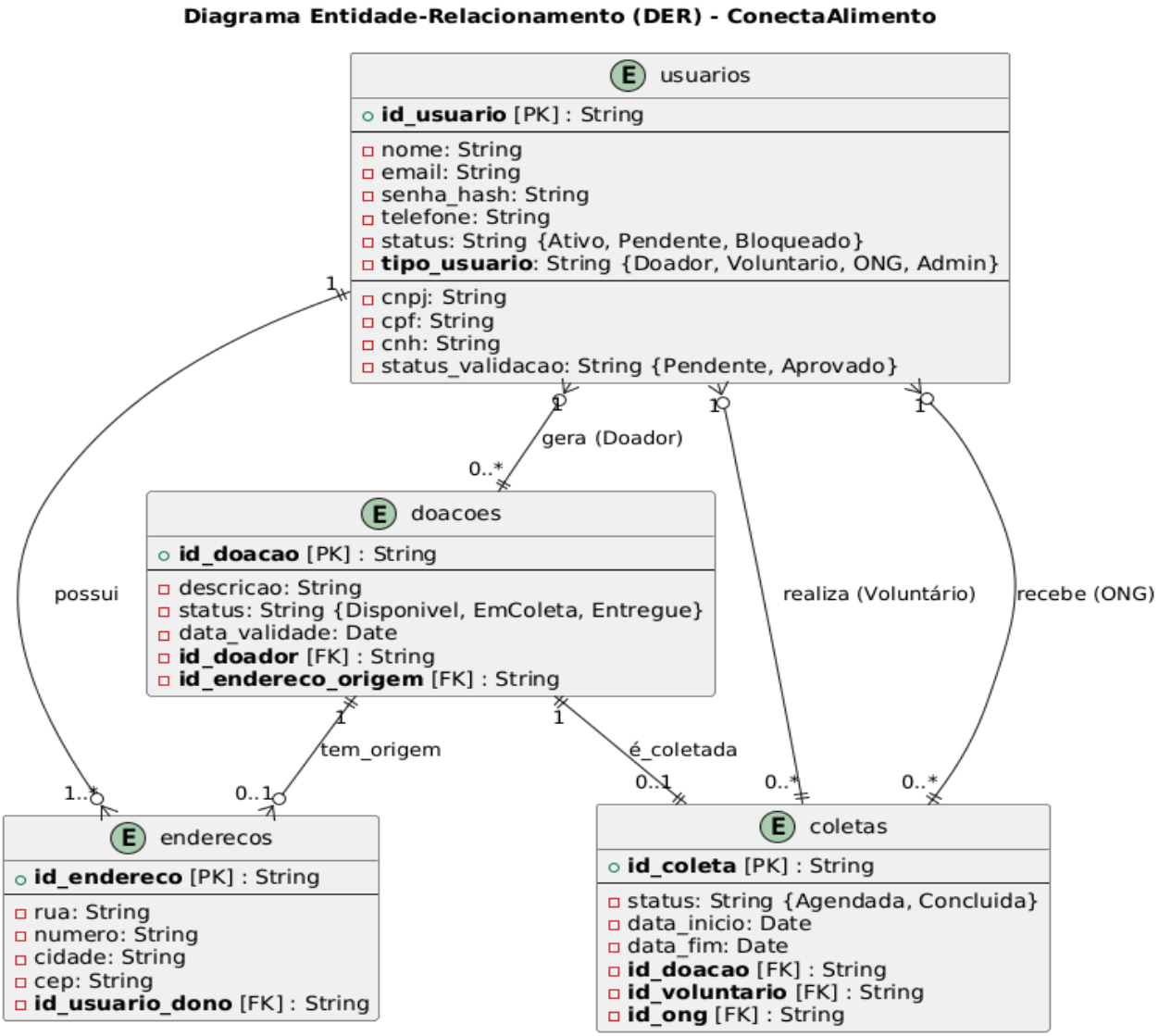
Diagrama de Estados - Ciclo de Vida do Usuário



Código PlantUML:

```
@startuml
title Diagrama de Estados - Ciclo de Vida do Usuário (Cadastro/Validação)

state "Pendente"
state "Ativo"
state "Rejeitado"
state "Bloqueado"

[*] --> Pendente : cadastrar() [ONG/Voluntário]

Pendente --> Ativo : validarCadastro(true) [Admin]
Pendente --> Rejeitado : validarCadastro(false) [Admin]

Ativo --> Bloqueado : bloquearUsuario() [Admin]
Bloqueado --> Ativo : desbloquearUsuario() [Admin]

Rejeitado --> [*]
@enduml
```

4. Modelos de Dados



Código PlantUML:

```
@startuml
title Diagrama Entidade-Relacionamento (DER) - ConectaAlimento

entity "usuarios" as T_USUARIO {
  + **id_usuario** [PK] : String
  --
  - nome: String
  - email: String
  - senha_hash: String
  - telefone: String
```

```
  - status: String {Ativo, Pendente, Bloqueado}
  - **tipo_usuario**: String {Doador, Voluntario, ONG, Admin}
  --
  - cnpj: String
  - cpf: String
  - cnh: String
  - status_validacao: String {Pendente, Aprovado}
}

entity "enderecos" as T_ENDERECO {
  + **id_endereco** [PK] : String
  --
  - rua: String
  - numero: String
  - cidade: String
  - cep: String
  - **id_usuario_dono** [FK] : String
}

entity "doacoes" as T_DOACAO {
  + **id_doacao** [PK] : String
  --
  - descricao: String
  - status: String {Disponivel, EmColeta, Entregue}
  - data_validade: Date
  - **id_doador** [FK] : String
  - **id_endereco_origem** [FK] : String
}

entity "coletas" as T_COLETA {
  + **id_coleta** [PK] : String
  --
  - status: String {Agendada, Concluida}
  - data_inicio: Date
  - data_fim: Date
  - **id_doacao** [FK] : String
  - **id_voluntario** [FK] : String
  - **id_ong** [FK] : String
}

T_USUARIO "1" ||--o{ "1..*" T_ENDERECO : "possui"
T_USUARIO "1" }o--|| "0..*" T_DOACAO : "gera (Doador)"
T_USUARIO "1" }o--|| "0..*" T_COLETA : "realiza (Voluntário)"
T_USUARIO "1" }o--|| "0..*" T_COLETA : "recebe (ONG)"
```

T_DOACAO "1" ||--o{ "0..1" T_ENDERECO : "tem_origem"
T_DOACAO "1" ||--|| "0..1" T_COLETA : "é_coletada"
@enduml