

Trabalho Final de Técnicas

Fábio Rocha da Costa
Ariane Figueiredo

Documentação do Software de Leilão

O sistema possui algumas limitações:

Campo de data tem formato com máscara em inglês yyyy-mm-dd;

Campo de hora tem formato com mascara para HH:mm;

Campo monetário tem formato apenas separado com vírgula para centavos -

####,## após o evento de perda de foco o campo aplicará a máscara correta;

Ao criar um lance, a tela não vai atualizar automaticamente o lance vencedor e usuário que fez o melhor lance, somente após fechar e abrir novamente os valores estarão atualizados.

O software foi desenvolvido em Java Swing.

Foram utilizados os Padrões Domain Model, DAO, Singleton e Facade conforme definido abaixo:

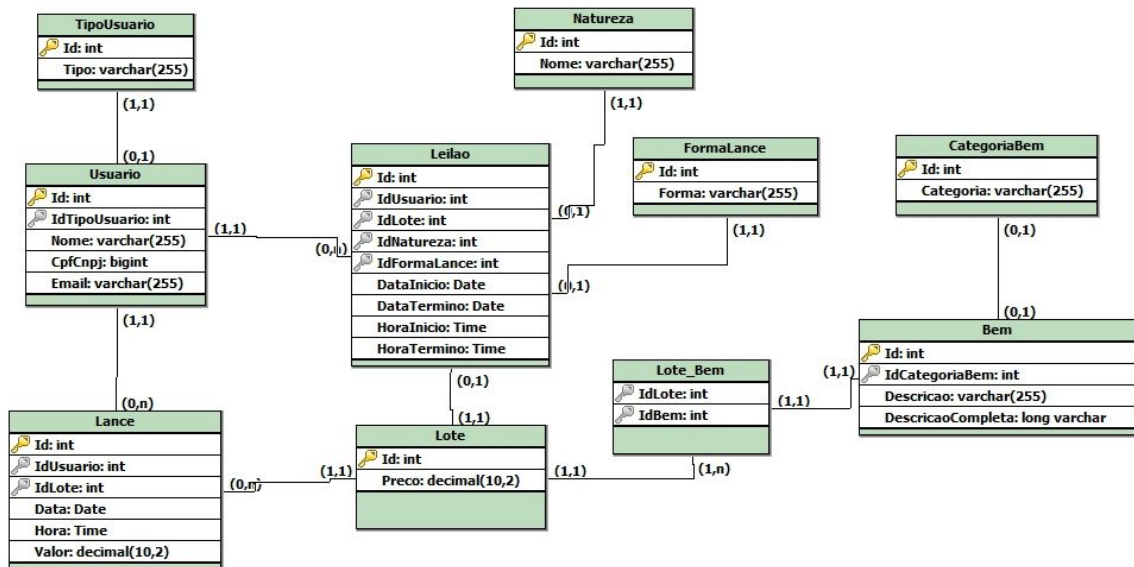
- Domain model:
 - Pacote Domain com classes de objetos do domínio;
- DAO:
 - Pacote Dao com metodos de persistência;
- Singleton:
 - Pacote Connection, classe DBConnection;
- Facade:
 - Pacote Facade;

Pacote Business possui regras de validação de formulários do negócio;

Foram criadas classes de exceção no pacote Exception.

Possui camada de apresentação definida no pacote View.

Utilizando o Derby para persistência de dados conforme modelo ER da imagem.



The diagram illustrates the Facade pattern for a botanical garden system, organized into three main layers: Facade, Domain, and DAO.

Facade Layer

- TipoUsuarioFacade**:
 - + TipoUsuarioFacade()
 - + buscarTodos() Collection<TipoUsuario>
- NaturezaFacade**:
 - + NaturezaFacade()
 - + buscarTodos() Collection<Natureza>
- UsuarioFacade**:
 - + UsuarioFacade()
 - + buscarTodos() Collection<Usuario>
 - + buscarPorId(int) Usuario
 - + buscarPorId(int) Usuario
 - + cancelarUsuario(Usuario) boolean
- LanceFacade**:
 - + LanceFacade()
 - + buscarTodosPorId(int) Collection<Lance>
 - + buscarPorIdPorId(int) int idNatureza int Lance
 - + cancelarLance(int) int
 - + cancelarLance(int) int
- BemFacade**:
 - + BemFacade()
 - + buscarTodos() Collection<Bem>
 - + buscarPorIdPorId(int) Collection<Bem>
 - + cancelarBem(Bem) int
 - + buscarPorIdPorId(int) Collection<Bem>
- LeliaoFacade**:
 - + LeliaoFacade()
 - + buscarPorId(int) Leliao
 - + buscarPorIdPorId(int) Collection<Leliao>
 - + cancelarLeliao(Leliao) boolean
- CategoriaBemFacade**:
 - + CategoriaBemFacade()
 - + buscarTodos() Collection<CategoriaBem>
- FormalanceFacade**:
 - + FormalanceFacade()
 - + buscarTodos() Collection<Formalance>

Domain Layer

- Lote**:
 - id Integer
 - + Lote() Lote() Integer Integer Integer Collection<Lance> bens Collection<Bem> preco BigDecimal
 - + getId() Integer
 - + setId(int) Integer void
 - + getPreco() BigDecimal
 - + setPreco(BigDecimal) void
 - + getLote() Lote
 - + setLote(Lote) void
 - + getLances() Collection<Lance>
 - + setLances(Collection<Lance>) void
 - + getBens() Collection<Bem>
 - + setBens(Collection<Bem>) void
- Bem**:
 - id Integer
 - descricao String
 - descricaoCompleta String
 - + Bem() Bem() Integer descricao String descricaoCompleta String categoriaBem CategoriaBem
 - + getId() Integer
 - + setId(int) Integer void
 - + getDescricao() String
 - + setDescricao(descricao String) void
 - + getDescricaoCompleta() String
 - + setDescricaoCompleta(descricaoCompleta String) void
 - + getLote() Lote
 - + setLote(Lote) void
 - + getCategoriaBem() CategoriaBem
 - + setCategoriaBem(categoriaBem CategoriaBem) void
 - + toString() String
- Leliao**:
 - id Integer
 - leliao() Leliao() Leliao() Integer Usuario Usuario Formalance Formalance Lote Lote Natureza Natureza dataInicio Date dataTermino Time horaTermino Time
 - + getId() Integer
 - + setId(int) Integer void
 - + getDateInicio() Date
 - + setDateInicio(dataInicio Date) void
 - + getDateTermino() Date
 - + setDateTermino(dataTermino Date) void
 - + getFormalance() Formalance
 - + setFormalance(formalance Formalance) void
 - + getLote() Lote
 - + setLote(Lote) void
 - + getNatureza() Natureza
 - + setNatureza(natureza Natureza) void
 - + getUsuario() Usuario
 - + setUsuario(usuario Usuario) void
 - + getHoraTermino() Time
 - + setHoraTermino(horaTermino Time) void
 - + getDateTermino() Time
 - + setDateTermino(dataTermino Time) void
- Lance**:
 - id Integer
 - lance() Lance() Integer data Date hora Time valor BigDecimal lote Lote usuario Usuario
 - + getId() Integer
 - + setId(int) Integer void
 - + getDate() Date
 - + setDate(data Date) void
 - + getValor() BigDecimal
 - + setValor(valor BigDecimal) void
 - + getLote() Lote
 - + setLote(lote Lote) void
 - + getUsuario() Usuario
 - + setUsuario(usuario Usuario) void
 - + getHoraTime() Time
 - + setHoraTime(hora Time) void
 - + toString() String
- CategoriaBem**:
 - id Integer
 - categoria String
 - + CategoriaBem() CategoriaBem() Integer categoria String
 - + getId() Integer
 - + setId(int) Integer void
 - + getCategoria() String
 - + setCategoria(categoria String) void
- Natureza**:
 - id Integer
 - nome String
 - + Natureza() Natureza() Integer nome String
 - + getId() Integer
 - + setId(int) Integer void
 - + getNome() String
 - + setNome(nome String) void
- Formalance**:
 - id Integer
 - forma String
 - + Formalance() Formalance() Integer forma String
 - + getId() Integer
 - + setId(int) Integer void
 - + getForma() String
 - + setForma(forma String) void
- Usuario**:
 - id Integer
 - nome String
 - cpf String
 - email String
 - tipoUsuario TipoUsuario
 - + Usuario() Usuario() Integer nome String cpf String email String tipoUsuario TipoUsuario
 - + getId() Integer
 - + setId(int) Integer void
 - + getNome() String
 - + setNome(nome String) void
 - + getCpf() String
 - + setCpf(cpf String) void
 - + getEmail() String
 - + setEmail(email String) void
 - + getTipoUsuario() TipoUsuario
 - + setTipoUsuario(tipoUsuario TipoUsuario) void
 - + toString() String
- TipoUsuario**:
 - id Integer
 - test String
 - + TipoUsuario() TipoUsuario() Integer test String
 - + getId() Integer
 - + setId(int) Integer void
 - + getTest() String
 - + setTest(test String) void

DAO Layer

- DAOBem**:
 - + getALL() Collection<Bem>
 - + insertBem(Bem) int
 - + getALLPorId(Collection<Bem>
 - + getALLPorId(Bem int) Collection<Bem>
 - + getALLPorId(int) Collection<Bem>
- DAOLote**:
 - + insert(Lote) int
 - + getALLPorId(int) Lote
- DAOCategoriaBem**:
 - + insert(CategoriaBem) int
 - + getALLPorId(int) CategoriaBem
- DAONatureza**:
 - + getALL() Collection<Natureza>
 - + getALLPorId(int) Natureza
- DAOFormalance**:
 - + getALL() Collection<Formalance>
 - + getALLPorId(int) Formalance
- DAOUsuario**:
 - + getALL() Collection<Usuario>
 - + insertUsuario(Usuario) boolean
 - + getALLPorIdPorId(int) Collection<Usuario>
 - + getALLPorId(int) Usuario
- DAOLance**:
 - + insert(Lance) int
 - + getALLPorId(int) Collection<Lance>
 - + getALLPorIdPorId(int idNatureza int) Lance
 - + remove(Lance int) void