Relatório de Modificações - CaseBem

Resumo das Alterações desde o Commit da Otávia (0ae31f6)

Este documento detalha todas as modificações realizadas no projeto CaseBem desde o último commit da Otávia ("correcoes do maroquio; usuario e casal com todos os testes passando") até o estado atual.

Histórico de Commits

1. **0ae31f6** - Otávia - "correcoes do maroquio; usuario e casal com todos os testes passando" (ponto de partida)
2. **5946663** - Ricardo Maroquio - "início da simplificacao"
3. **898eddf** - Ricardo Maroquio - "atualizacoes nos itens de demanda"
4. **1d6f5ed** - Ricardo Maroquio - "renomeacoes e acertos de bugs"
5. **b2ebd3c** - Ricardo Maroquio - "refafotarcao completa"
6. **f5157da** - Ricardo Maroquio - "projeto atualizado com novo esquema de classes e com testes funcionando totalmente"

Principais Mudanças Estruturais

1. Simplificação do Modelo de Usuários

- **Removido**: Modelos separados para `Noivo`, `Administrador`, `Fornecedor` e `Prestador`
- **Mantido**: Apenas o modelo `Usuario` com campo `tipo` para diferenciar os papéis
- **Impacto**: Simplificação significativa da estrutura, eliminando herança desnecessária

2. Reestruturação de Itens de Contrato

- **Renomeado**: `ItemContrato` → `ItemDemanda`
- **Criado**: Separação entre `ItemDemandaProduto` e `ItemDemandaServico`
- **Motivo**: Melhor representação do domínio onde demandas precedem contratos

3. Implementação Completa de Orçamentos

- **Criado**: Sistema completo de orçamentos com estrutura similar a demandas
- **Componentes**:

    - `model/orcamento_model.py`
    - `model/item_orcamento_produto_model.py`
    - `model/item_orcamento_servico_model.py`
    - Repositórios e SQLs correspondentes
    - Testes completos para todas as funcionalidades

Detalhamento das Modificações

Models (`/model/`)

Removidos:

- `administrador_model.py`
- `fornecedor_model.py`
- `prestador_model.py`
- `noivo_model.py` (nunca existiu como arquivo separado)

Modificados:

- **`usuario_model.py`**:
    - Adicionado campo `tipo` (ENUM: ADMIN, NOIVO, FORNECEDOR, PRESTADOR)
    - Centralização de todos os tipos de usuário em uma única entidade
- **`casal_model.py`**:
    - Simplificado para referenciar IDs de usuários do tipo NOIVO
    - Removida herança complexa
- **`demanda_model.py`**:
    - Renomeado conceito de "contrato" para "demanda"
    - Representa lista de desejos do casal

Novos:

- **`item_demanda_produto_model.py`**: Item de demanda para produtos
- **`item_demanda_servico_model.py`**: Item de demanda para serviços
- **`orcamento_model.py`**: Orçamento principal com status (PENDENTE, ACEITO, REJEITADO)
- **`item_orcamento_produto_model.py`**: Item de orçamento para produtos com preço
- **`item_orcamento_servico_model.py`**: Item de orçamento para serviços com preço

SQL (`/sql/`)

Modificações Principais:

- **Tabelas de usuário unificadas**: Removidas tabelas específicas por tipo
- **Foreign Keys**: Todas apontam para tabela `usuario`
- **Novos comandos SQL para orçamentos**:
    - Query especial `ACEITAR_ORCAMENTO_E_REJEITAR_OUTROS`
    - Cálculo de totais por orçamento

Repositórios (`/repo/`)

Removidos:

- `administrador_repo.py`
- `fornecedor_repo.py`
- `prestador_repo.py`

Modificados:

- **`usuario_repo.py`**: Centraliza todas operações de usuário
- **`casal_repo.py`**: Adicionado método `obter_casal_por_noivo`

Novos:

- **`orcamento_repo.py`**:
    - Método especial `aceitar_orcamento_e_rejeitar_outros`
    - Métodos de busca por status, demanda, fornecedor/prestador
- **`item_orcamento_produto_repo.py`**:
    - Inclui `calcular_total_itens_produto_orcamento`
- **`item_orcamento_servico_repo.py`**:
    - Inclui `calcular_total_itens_servico_orcamento`

Testes (`/tests/`)

Estrutura de Fixtures (`conftest.py`):

- **Problema corrigido**: Conflito de emails e telefones únicos entre diferentes tipos de usuários
- **Solução**:
    - Noivos: emails `usuario{i}@email.com`, telefones `(28) 99999-00{i}`
    - Prestadores: emails `prestador{i}@email.com`, telefones `(28) 99999-20{i}`
    - Fornecedores: emails `fornecedor{i}@email.com`, telefones `(28) 99999-30{i}`
    - Administradores: emails `admin{i}@email.com`, telefones `(28) 99999-40{i}`

Novos Testes:

- `test_orcamento_repo.py`: 12 testes cobrindo todas funcionalidades
- `test_item_orcamento_produto_repo.py`: 9 testes incluindo cálculo de totais
- `test_item_orcamento_servico_repo.py`: 9 testes incluindo cálculo de totais

Correções de Bugs

1. Foreign Key Constraints

- **Problema**: Ordem incorreta de criação de tabelas
- **Solução**: Sempre criar tabelas de dependência antes (usuario → casal → demanda → itens)

2. Unique Constraints

- **Problema**: Emails e telefones duplicados entre fixtures
- **Solução**: Namespaces diferentes para cada tipo de usuário

3. Nomenclatura Inconsistente

- **Problema**: Mistura entre "ItemContrato" e "ItemDemanda"
- **Solução**: Padronização completa para "ItemDemanda"

Estado Final do Projeto

Estatísticas:

- **Total de testes**: 122 (todos passando)
- **Cobertura**: Todos os repositórios com testes completos
- **Warnings**: Apenas deprecation warnings do datetime adapter do SQLite

Arquitetura:

```
CaseBem/
├── model/          # Modelos de dados (dataclasses)
├── sql/            # Comandos SQL
├── repo/           # Repositórios (camada de acesso a dados)
├── tests/          # Testes unitários
│   └── conftest.py # Fixtures compartilhadas
└── util/           # Utilitários (conexão com BD)
```

Fluxo de Negócio:

1. **Usuários** se cadastram com tipos específicos
2. **Casais** são formados por dois noivos
3. **Demandas** são criadas pelos casais com itens desejados
4. **Fornecedores/Prestadores** enviam orçamentos para as demandas

5. **Casais** podem aceitar orçamentos (rejeitando automaticamente os demais)