Projeto Aplicado em Desenvolvimento de Sistemas

Aluno: Vinícius Neris Ferreira Santana

Matrícula: 202210326

Professor: Wanderson Pereira dos Santos

Disciplina: Projeto Aplicado em Desenvolvimento de Sistemas

Módulo Financeiro - Documentação

1. Objetivo do Módulo Financeiro

Gerenciar todos os aspectos financeiros do sistema de gestão de eventos fotográficos, incluindo pedidos, faturas, formas de pagamento e comprovantes. O módulo oferece visões diferentes para gerente, funcionário e cliente, possibilitando controle, registro e auditoria das transações.

2. Funcionalidades

• **Pedidos/Serviços:** registrar cada serviço ou venda realizada;
• **Faturas:** calcular valor total, gerar fatura vinculada ao pedido e apresentar status (pago/pendente);
• **Forma de pagamento:** suportar PIX, cartão e boleto (campo indicando a forma);
• **Comprovante:** permitir upload de comprovantes ou registro de número de transação;
• **Relatórios:** geração de relatórios mensais consolidados por fotógrafo e por evento.

3. Perfis de Acesso

Gerente: visão completa do módulo: visualiza todos os pedidos, faturas, relatórios e pode marcar pagamentos como conciliados.
Funcionário: cria pedidos, registra pagamentos, verifica comprovantes e interage com faturas pendentes.
Cliente: consulta suas faturas, visualiza detalhes do pedido e envia comprovante de pagamento.

4. Estrutura do Banco de Dados (Resumo)

Tabelas principais:
- **pedidos**: id, cliente_id, descricao, valor_total, status, created_at
- **faturas**: id, pedido_id, valor, status_pagamento, vencimento, created_at
- **pagamentos**: id, fatura_id, forma_pagamento, valor_pago, comprovante_path, data_pagamento
- **usuarios**: id, nome, email, role (gerente/funcionario/cliente), senha_hash

Scripts SQL básicos estão descritos abaixo e devem ser colocados em `_BD/esquema.sql`.

5. Script SQL - Tabelas

```sql
CREATE TABLE usuarios (
id INTEGER PRIMARY KEY AUTOINCREMENT,
nome TEXT NOT NULL,
email TEXT UNIQUE NOT NULL,
role TEXT CHECK(role IN ('gerente','funcionario','cliente')) NOT NULL,
senha_hash TEXT NOT NULL
);

CREATE TABLE pedidos (
id INTEGER PRIMARY KEY AUTOINCREMENT,
cliente_id INTEGER NOT NULL,
descricao TEXT,
valor_total REAL NOT NULL,
status TEXT CHECK(status IN ('aberto','faturado','cancelado')) NOT NULL DEFAULT 'aberto',
created_at TEXT DEFAULT CURRENT_TIMESTAMP,
FOREIGN KEY(cliente_id) REFERENCES usuarios(id)
);

CREATE TABLE faturas (
id INTEGER PRIMARY KEY AUTOINCREMENT,
pedido_id INTEGER NOT NULL,
valor REAL NOT NULL,
status_pagamento TEXT CHECK(status_pagamento IN ('pendente','pago')) DEFAULT 'pendente',
vencimento TEXT NOT NULL,
created_at TEXT DEFAULT CURRENT_TIMESTAMP,
FOREIGN KEY(pedido_id) REFERENCES pedidos(id)
);

CREATE TABLE pagamentos (
id INTEGER PRIMARY KEY AUTOINCREMENT,
fatura_id INTEGER NOT NULL,
forma_pagamento TEXT NOT NULL,
valor_pago REAL NOT NULL,
comprovante_path TEXT,
data_pagamento TEXT DEFAULT CURRENT_TIMESTAMP,
FOREIGN KEY(fatura_id) REFERENCES faturas(id)
);
```

6. Endpoints Principais (Backend)

POST /api/pedidos - Criar um novo pedido.
GET /api/pedidos - Listar pedidos (filtros por status, cliente e período).
GET /api/pedidos/:id - Detalhar pedido.
POST /api/faturas - Gerar fatura para um pedido.
POST /api/pagamentos - Registrar pagamento (aceita upload de comprovante).
GET /api/relatorios/financeiro?mes=MM&ano;=YYYY - Relatório mensal para gerente.

7. Fluxo Resumido (Exemplo)

1. Funcionário cria um pedido no sistema e associa a um cliente.
2. Sistema gera fatura vinculada ao pedido (com data de vencimento).
3. Cliente realiza pagamento externamente (PIX/cartão) e faz upload do comprovante.
4. Funcionário valida o comprovante e registra o pagamento no sistema.
5. Gerente consulta relatórios mensais e realiza conciliação.