Relatório de Desenvolvimento do Sistema de Gerenciamento de Produtos

1. Introdução

Este relatório descreve o desenvolvimento de um sistema de gerenciamento de produtos, onde os usuários podem criar, ler, atualizar e excluir (CRUD) informações sobre produtos. O sistema é dividido em duas partes principais: a API em PHP que interage com o banco de dados e a interface web em HTML/JavaScript que permite a interação do usuário.

2.1. Banco de Dados

Um banco de dados chamado "api" foi criado, contendo uma tabela chamada "cliente" com os seguintes campos:

- `idcli` (INT, auto-incremento, chave primária)
- `nome` (VARCHAR)
- `descricao` (VARCHAR)
- `precoVenda` (DECIMAL)
- `precoCusto` (DECIMAL)
- `lucro` (DECIMAL)
- `quantidade` (INT)
- `categoria` (VARCHAR)

2.2. API em PHP

A API foi desenvolvida em PHP para gerenciar as operações CRUD. Abaixo estão as etapas detalhadas da implementação da API:

2.2.1. Configuração do Banco de Dados

php
// Configurações do banco de dados
$servername = "localhost";
$username = "root";
$password = "";

// Criação da conexão e do banco de dados

```
$conn = new mysqli($servername, $username, $password);
$conn->query("DROP DATABASE IF EXISTS api");
$conn->query("CREATE DATABASE api");
$conn->select_db("api");

// Criação da tabela 'cliente'
$sql = "CREATE TABLE cliente (
    idcli INT(11) AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(255) NOT NULL,
    descricao VARCHAR(255) NOT NULL,
    precoVenda DECIMAL(10, 2) NOT NULL,
    precoCusto DECIMAL(10, 2) NOT NULL,
    lucro DECIMAL(10, 2) NOT NULL,
    quantidade INT NOT NULL,
    categoria VARCHAR(255) NOT NULL
)";
$conn->query($sql);
```

2.2.2. Implementação da API

O código da API foi organizado para lidar com as diferentes operações baseadas no método HTTP (GET, POST, PUT, DELETE):

- **GET**: Para recuperar a lista de produtos.
- **POST**: Para adicionar um novo produto.
- **PUT**: Para atualizar um produto existente.
- **DELETE**: Para excluir um produto.

2.3. Interface Web

A interface web foi construída em HTML com JavaScript para permitir a interação do usuário com a API. O código HTML/JavaScript contém os seguintes elementos:

- Campos de entrada para nome, descrição, preços, lucro, quantidade e categoria.
- Botões para executar as operações CRUD.
- Uma tabela que exibe os produtos cadastrados.

2.4. Cálculo do Lucro

Um evento foi adicionado para calcular automaticamente o lucro com base nos valores de preço de venda e preço de custo. O lucro é exibido em um campo somente leitura.

3. Código Completo

3.1. API em PHP

```php
<?php
$uri = parse_url($_SERVER['REQUEST_URI'], PHP_URL_PATH);
$uri = explode('/', $uri);

$response = array();
$response['method'] = $_SERVER['REQUEST_METHOD'];

$servername = "localhost";
$username = "root";
$password = "";
$dbname = "api";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect_error) {
    die("Erro na conexão com o banco de dados: " . $conn->connect_error);
}

switch ($_SERVER['REQUEST_METHOD']) {
    case 'PUT':
        $data = json_decode(file_get_contents("php://input"), true);
        $id = $uri[4];

        $stmt = $conn->prepare("UPDATE cliente SET nome=?, descricao=?, precoVenda=?,
precoCusto=?, lucro=?, quantidade=?, categoria=? WHERE idcli=?");
        $stmt->bind_param("ssddidsi", $data['nome'], $data['descricao'], $data['precoVenda'],
$data['precoCusto'], $data['lucro'], $data['quantidade'], $data['categoria'], $id);

        if ($stmt->execute()) {
            $response['message'] = $stmt->affected_rows > 0 ? "Cliente atualizado com
sucesso!" : "Nada foi alterado!";
        } else {
            $response['message'] = "Erro ao atualizar cliente: " . $conn->error;
        }
        break;

    case 'POST':
        $data = json_decode(file_get_contents("php://input"), true);

        $stmt = $conn->prepare("INSERT INTO cliente (nome, descricao, precoVenda,
precoCusto, lucro, quantidade, categoria) VALUES (?, ?, ?, ?, ?, ?, ?)");
        $stmt->bind_param("ssddids", $data['nome'], $data['descricao'], $data['precoVenda'],
$data['precoCusto'], $data['lucro'], $data['quantidade'], $data['categoria']);

        if ($stmt->execute()) {
            $response['message'] = "Cliente adicionado com sucesso!";
```

```
        } else {
            $response['message'] = "Erro ao adicionar cliente: " . $conn->error;
        }
        break;

    case 'DELETE':
        $id = $uri[4];
        $stmt = $conn->prepare("DELETE FROM cliente WHERE idcli=?");
        $stmt->bind_param("i", $id);

        if ($stmt->execute()) {
            $response['message'] = $stmt->affected_rows > 0 ? "Cliente excluído com sucesso!"
: "Erro ao tentar excluir cliente.";
        } else {
            $response['message'] = "Erro ao tentar excluir cliente: " . $conn->error;
        }
        break;

    case 'GET':
        $sql = "SELECT * FROM cliente";
        $result = $conn->query($sql);

        if ($result->num_rows > 0) {
            while ($row = $result->fetch_assoc()) {
                $response[] = [
                    'idcli' => $row['idcli'],
                    'nome' => $row['nome'],
                    'descricao' => $row['descricao'],
                    'precoVenda' => $row['precoVenda'],
                    'precoCusto' => $row['precoCusto'],
                    'lucro' => $row['lucro'],
                    'quantidade' => $row['quantidade'],
                    'categoria' => $row['categoria']
                ];
            }
        } else {
            $response['message'] = "Nenhum cliente encontrado.";
        }
        break;

    default:
        $response['message'] = "Método não suportado.";
}

$conn->close();
header('Content-Type: application/json');
echo json_encode($response);
?>
```

```

### 3.2. Interface HTML/JavaScript

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Teste de Requisições</title>
    <style>
        table {
            border-collapse: collapse;
            width: 100%;
        }
        th, td {
            border: 1px solid #dddddd;
            text-align: left;
            padding: 8px;
        }
        th {
            background-color: #f2f2f2;
        }
        tr:hover {
            background-color: #f5f5f5;
            cursor: pointer;
        }
    </style>
</head>
<body>
    <h1>Gerenciamento de Produtos</h1>

    <input type="hidden" id="idcli" value="0" />
    Nome: <input type="text" id="nome" /><br />
    Descrição: <input type="text" id="descricao" /><br />
    Preço de Venda: <input type="number" id="precoVenda" step="0.01" /><br />
    Preço de Custo: <input type="number" id="precoCusto" step="0.01" /><br />
    Lucro: <input type="number" id="lucro" step="0.01" readonly /><br />
    Quantidade: <input type="number" id="quantidade" /><br />
    Categoria: <input type="text" id="categoria" /><br /><br />

    <button onclick="get()">GET (Consultar)</button>
    <button onclick="post()">POST (Cadastrar)</button>
    <button onclick="put()">PUT (Alterar)</button>
    <button onclick="del()">DELETE (Excluir)</button>
    <div id="table-container"></div>
```
```

```
<script>
    async function calculateProfit() {
        const precoVenda = parseFloat(document.getElementById("precoVenda").value) || 0;
        const precoCusto = parseFloat(document.getElementById("precoCusto").value) || 0;
        document.getElementById("lucro").value = (precoVenda - precoCusto).toFixed(2);
    }

    document.getElementById("precoVenda").addEventListener("input", calculateProfit);
    document.getElementById("precoCusto").addEventListener("input", calculateProfit);

    async function
```