UNIVERSIDADE DO OESTE DE SANTA CATARINA – UNOESC
CAMPUS DE SÃO MIGUEL DO OESTE

DOUGLAS LUIZ SCHNEIDER
LEONARDO FRANZEN
SIDIMAR LUIZ HOELSCHER

RELATÓRIO MUSEU MBH

São Miguel do Oeste – Santa Catarina
2020

DOUGLAS LUIZ SCHNEIDER
LEONARDO FRANZEN
SIDIMAR LUIZ HOELSCHER

RELATÓRIO MUSEU MBH

Trabalho final das disciplinas Programação IV, Engenharia de Software II e Banco de Dados II apresentado ao curso Ciência da Computação da Universidade do Oeste de Santa Catarina como requisito para avaliação das três disciplinas.

Professores: Franciele Petry e Roberson Junior Fernandes Alves

São Miguel do Oeste
2020

LISTA DE ILUSTRAÇÃO

LISTA DE QUADROS

Sumário

1 INTRODUÇÃO

Para que o museu facilite o acesso aos visitantes e tenha um controle dos mesmos e também do seu acervo, tem a necessidade de construção e implementação de um sistema para realizar o gerenciamento do mesmo.

A ideia é que o possa ser possa cadastrar e demonstrar detalhes do acervo ao público. Também tem a necessidade de registrar as visitas feita por cada visitante, com data e horário.

2 DESENVOLVIMENTO

O projeto consiste em criar um software web para gerenciamento de museu. O software deve seguir requisitos exigidos pelas seguintes matérias e professores, Engenharia de Software e Programação IV, pela professora Franciele Petry, e Banco de Dados II, pelo professor Roberson Junior Fernandes Alves.

2.1 ENGENHARIA DE SOFTWARE

2.1.1 Requisitos

Para construção do software serão seguidos os seguintes requisitos:

Cadastrar informações sobre a instituição responsável pelo museu;
Cadastrar as informações do acervo do museu;
Possibilidade de categorizar os objetos do acervo;
Permitir anexar arquivos de imagem, documentos de texto, áudios e até mesmo vídeo para objetos do museu;
Os visitantes podem consultar dados dos objetos do museu;
Os funcionários dos museus, classificados conforme sua função, poderão cadastrar, ativar/desativar e controlar outras informações dos objetos do museu;
Possibilidade mudar o status dos objetos do museu, ou seja, certos objetos podem não estar mais em exposição, podem estar em manutenção, etc;
Registrar a presença de visitantes de forma a emitir relatórios de quantidade de pessoas, faixa etária, etc;

Quadro 1: Requisitos para a construção do sistema
Fonte: Os autores (2020)

Com isso, pode ser construídos os diagramas e o software será construído.

2.1.2 Diagramas

2.1.2.1 Diagramas de caso de uso

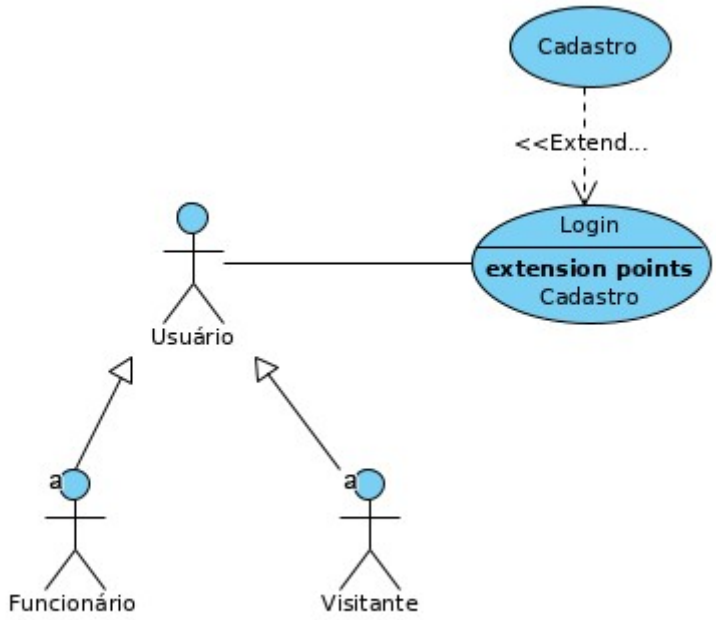

Figura 1: Diagrama de Login
Fonte: Os Autores (2020)

O diagrama da figura 1 mostra como é feita o login pelo usuário. Para efetuar o login, o usuário terá que ser um visitante ou funcionário, o sistema verifica se o mesmo está cadastrado e acessa o sistema.

O próximo diagrama informa como será o funcionamento do cadastro de visitante. O visitante acessa a página de registro, informa os dados necessário e o sistema processa e salva esses dados, isso é mostrado na figura a seguir.

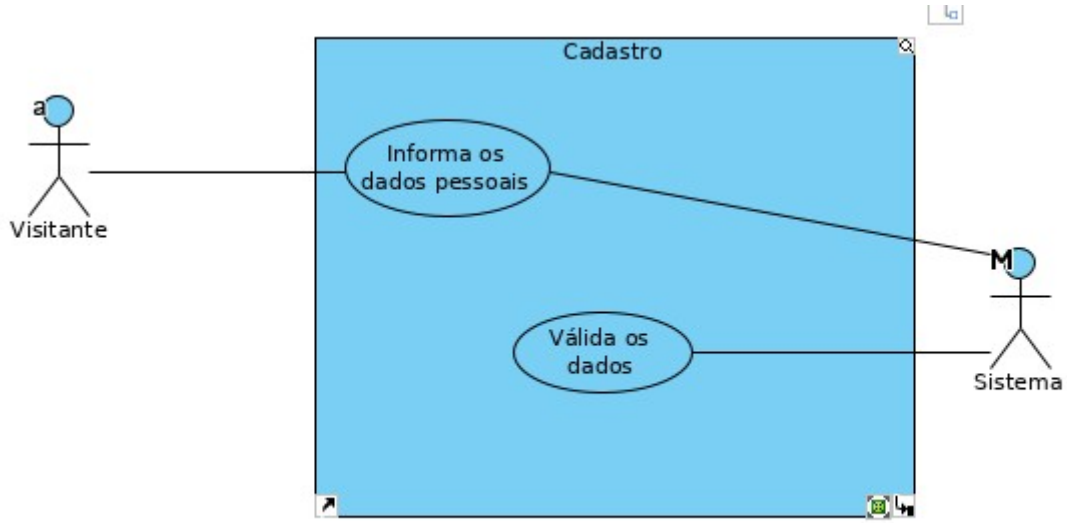

Figura 2: Cadastro de visitantes
Fonte: Os autores (2020)

Os visitantes podem fazer o agendamento de visitas, bastando apenas que tenham cadastrado realizado no sistema, como mostra a figura 3.

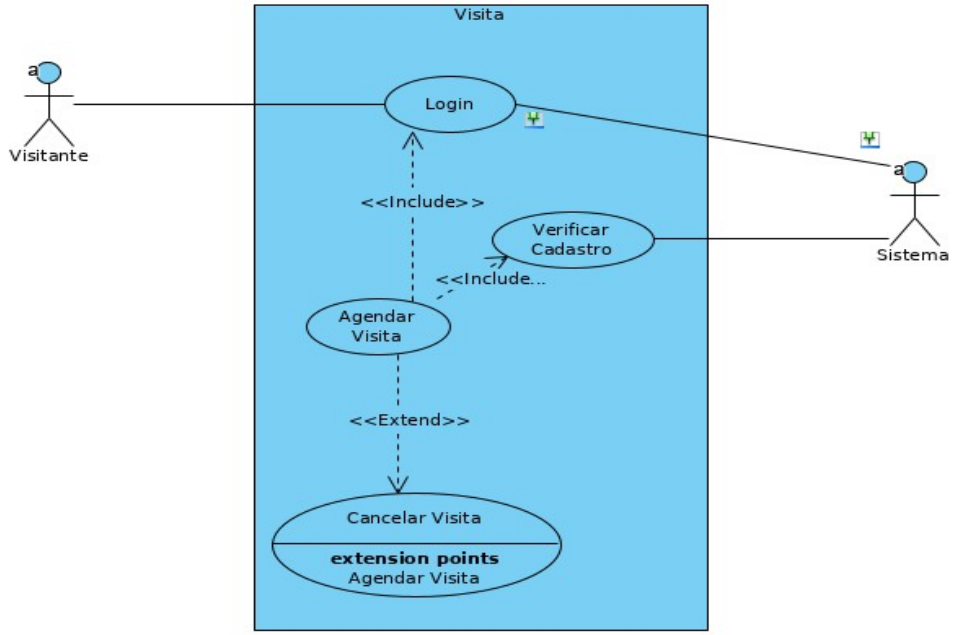

Figura 3: Agendar visitas
Fonte: Os autores (2020)

O usuário efetua o login, o sistema verifica se o mesmo possui cadastro, se possui, o mesmo pode agendar a visita e também pode cancelar a mesma.

O sistema também pode cadastrar funcionários, o usuário cadastrado acessa o sistema e o mesmo verifica, com isso preenche os dados necessário e informa o cargo, podendo dar acesso ao novo funcionário ou não. A figura abaixo mostra como é realizado o processo.

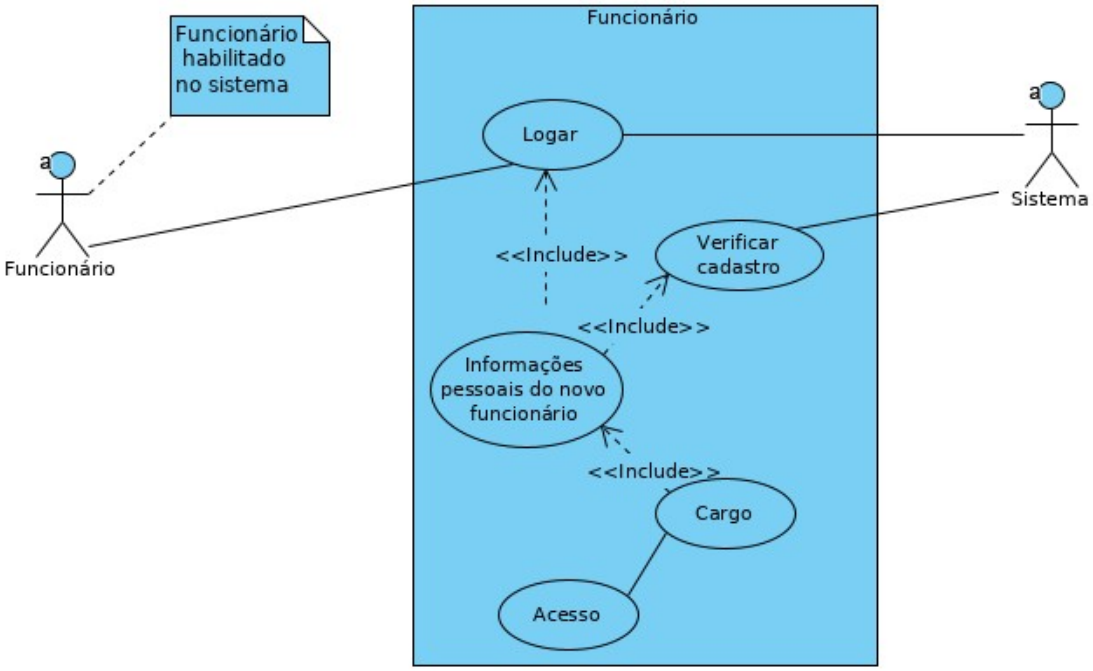

Figura 4: Cadastro de funcionário
Fonte: Os autores (2020)

O sistema também cadastra os objetos do museu. O funcionário recebe o objeto do proprietário e acessa o sistema para catalogar o objeto, informando a categoria e o status do mesmo, além de poder anexar arquivos a ele. A figura a seguir mostra como funciona o cadastro.

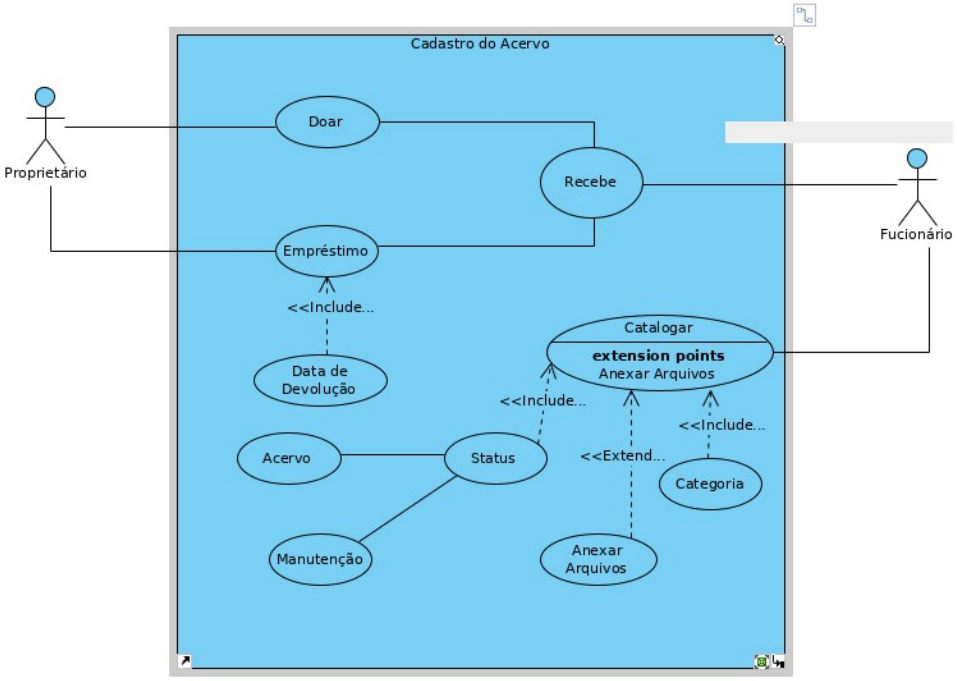

Figura 5: Cadastro de objetos
Fonte: Os autores (2020)

Além disso, o sistema é permitido alterar as informações do objeto no sistema. O funcionário cadastrado acessa e o sistema verifica seu cadastro. Com isso, o usuário pode alterar informações como o status, descrição e arquivos que o objeto possui para os visitantes acessarem, como é mostrado na figura abaixo.

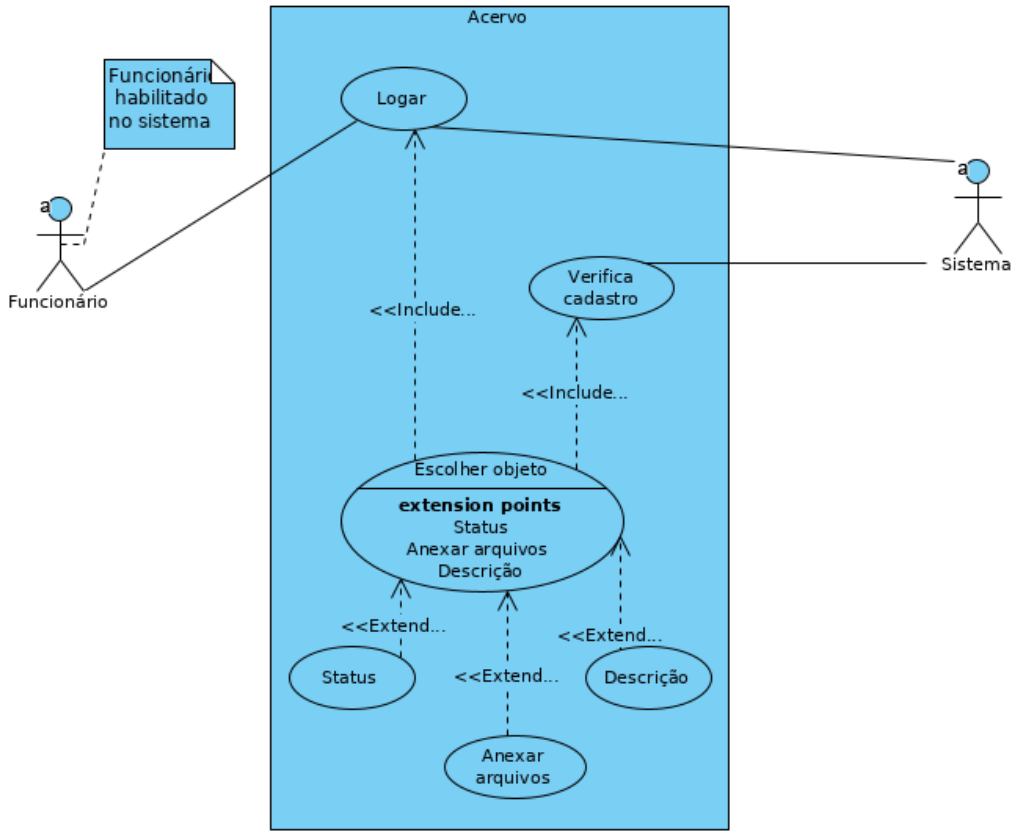

Figura 6: Alterar objetos
Fonte: Os autores (2020)

2.1.2.2 Diagrama de classe

O diagrama de classe demonstra o funcionamento do software para o museu. Possuindo todas as classes necessárias para a construção do sistema.

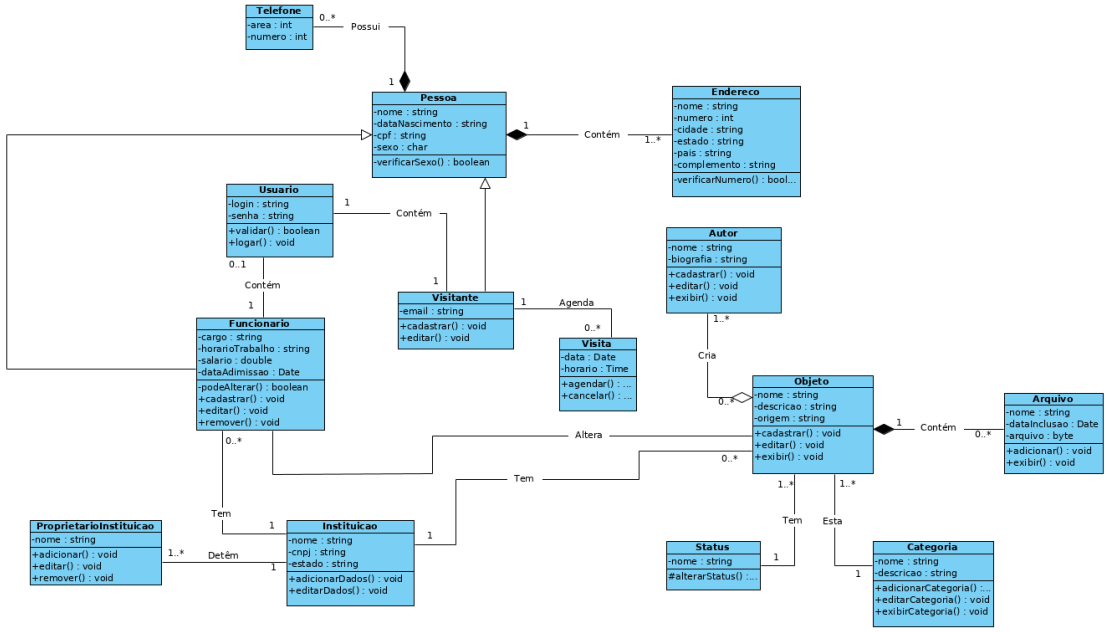

Figura 7: Diagrama de classes do museu
Fonte: Os autores (2020)

O diagrama contém a classe Pessoa, onde podemos informar os dados dela, como nome data de nascimento, CPF e sexo. Tendo uma função para verificar se o sexo é masculino ou feminino.

Com a classe pessoa temos o Endereço, onde é informado a rua, número, complemento, cidade, estado, país e a função para verificar se o número do endereço não é negativo.

Ainda junto com a classe Pessoa, tem a classe Telefone, onde é armazenado o código de área e o número de telefone.

Ligado a essa classe geral temo a classe Funcionario, onde há informações do funcionário da instituição, onde temos o cargo, horário de trabalho, salário e a data de admissão. As funções são para editar, cadastrar e remover o funcionário e também verificando se ele pode alterar um objeto do museu.

A também a classe Visitante, que é informado o e-mail, podendo cadastrar ou editar as informações do visitante. Atrelado a essa classe, tem a classe de Visita, onde é informado a data e o horário da visita, podendo agendar ou cancelar a mesma.

Conectado as duas temos a classe Usuario, que simplesmente tem um login e senha para ele. Contendo apenas as funções de logar e validar.

A classe Instituicao armazena os dados da própria instituição como nome, CNPJ e o estado onde esta se localiza. Esses dados são adicionados pela função adicionarDados e editados pela função editarDados. Junto a essa classe tem a classe ProprietarioInstituicao, para

armazenar o nome dos proprietários da instituição, podendo ser adicionado, editados e/ou removidos.

A classe Objeto contem toda descrição do objeto que se encontra no acervo do museu, como nome, descrição e origem, podendo ser cadastrado, editado e exibido.

Outras classes que são necessárias para complementar a classe Objeto são a classe Status, onde apenas informa o nome do status, com uma função em que pode ser alterada pela classe Objeto, a classe Categoria, onde tem o nome dessa categoria e sua descrição, podendo adicionar, editar ou exibir, a classe Arquivo, informando o nome, a data de inclusão e o arquivo em si, podendo ser adicionado e exibido, e a classe Autor, contendo alguns dados dos autores, podendo, também, ser cadastrado exibido ou editado.

2.1.2.3 Diagramas de atividades

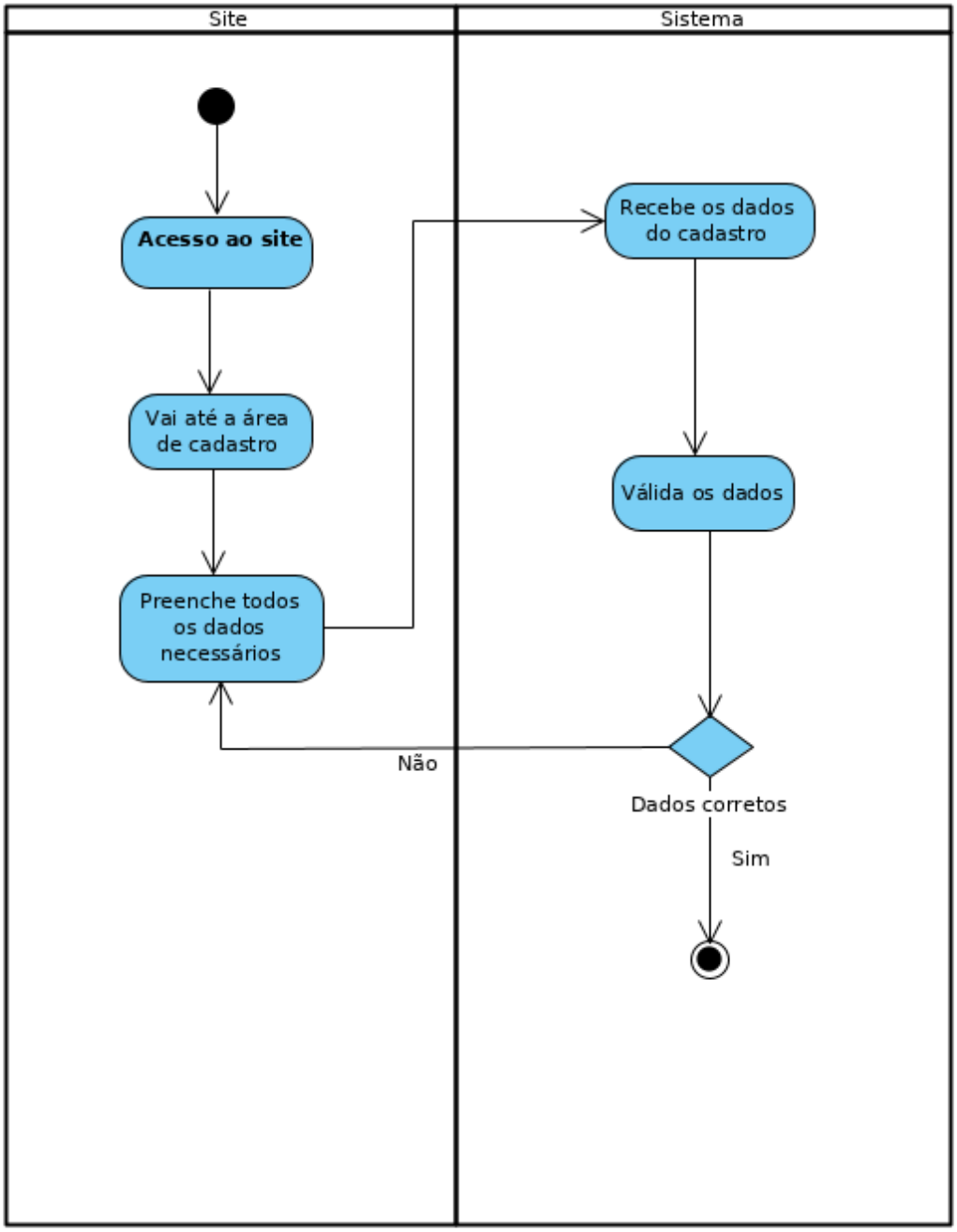

Figura 8: Cadastro de visitantes
Fonte: Os autores (2020)

Para o cadastro do visitante, o mesmo acessa o site e entra na área de cadastro, preenchendo todos os dados necessário, o sistema recebe esses dados e válida, se os dados estiverem corretos, o processo é finalizado, caso contrário, ele retorna para a página de cadastro para preencher os dados novamente.

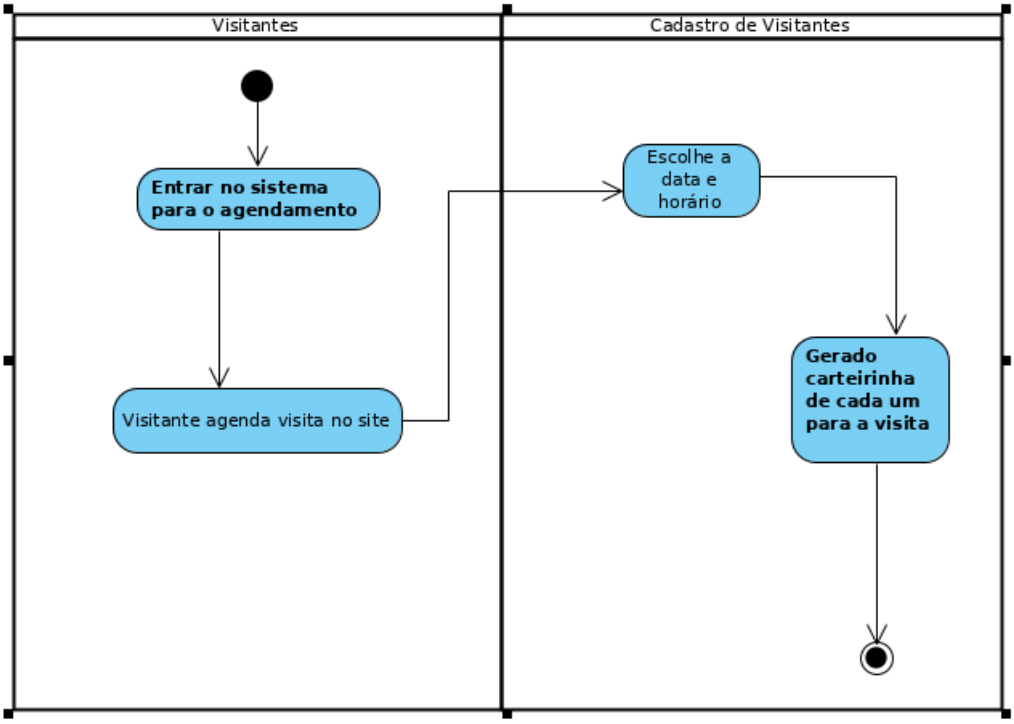

Figura 9: Agendamento de visitas
Fonte: Os autores (2020)

Para agendar a visita, o usuário acessa o sistema e agenda uma visita no site, escolhendo data e horário. Com isso o sistema grava e gera uma identificação dessa visita.

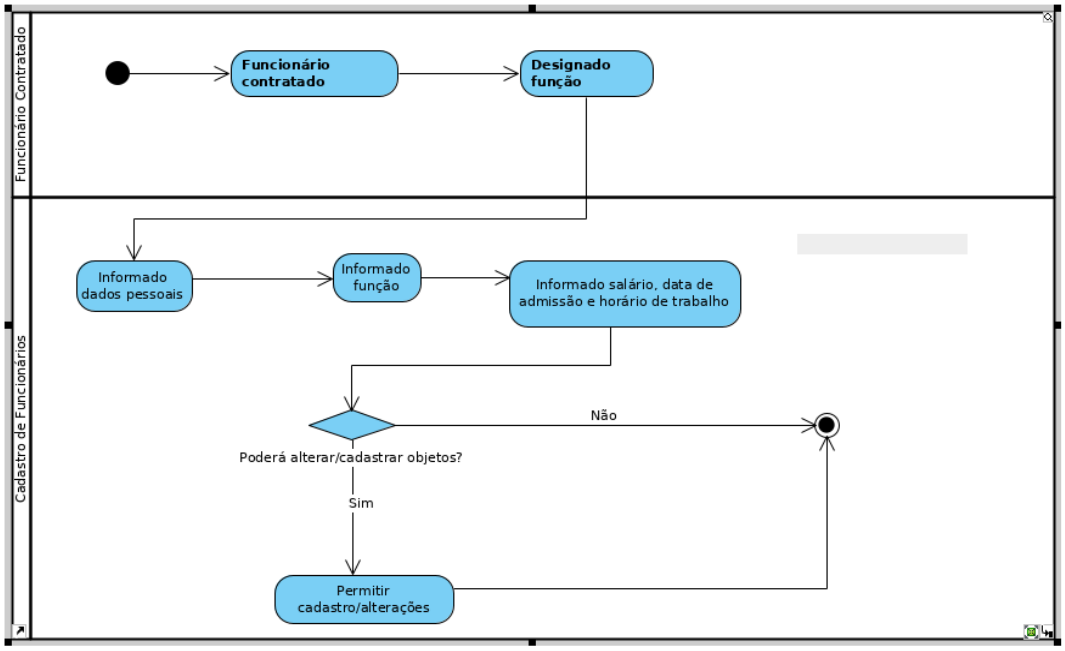

Figura 10: Cadastro de funcionário
Fonte: Os autores (2020)

No diagrama da figura 10, o novo funcionário tem sua função designada e com isso, para cadastro no sistema, são informados os dados pessoais, função, salário e data de admissão. Caso esse funcionário tenha acesso permissão para editar e cadastrar informações no sistema, a ele é concebido essa permissão e finalizado o processo, caso não seja dado essa permissão o processo é finalizado.

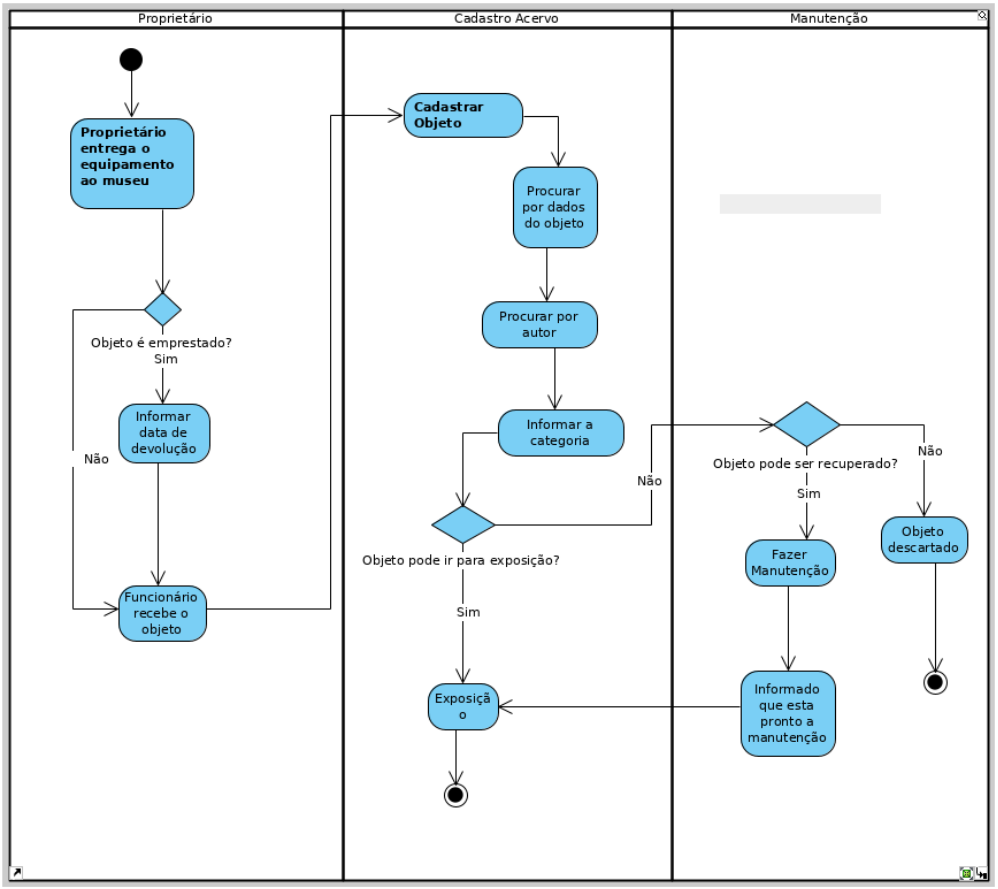

Figura 11: Cadastro de objeto
Fonte: Os autores (2020)

Começa pela parte do proprietário, onde ele entrega o objeto ao museu, informando se o mesmo é para doação ou um empréstimo, nesse caso é informado a data de devolução. O funcionário recebe o objeto e começa o cadastro, informando os dados, autores, a categoria do objeto e se o mesmo vai para a exposição ou manutenção. Caso ele vá para a exposição, o processo é finalizado. Caso o objeto vá para manutenção, ele pode ser descartado e, portanto, finalizado a atividade, ou feita a manutenção para retornar a exposição.

2.1.2.4 Diagramas de sequência

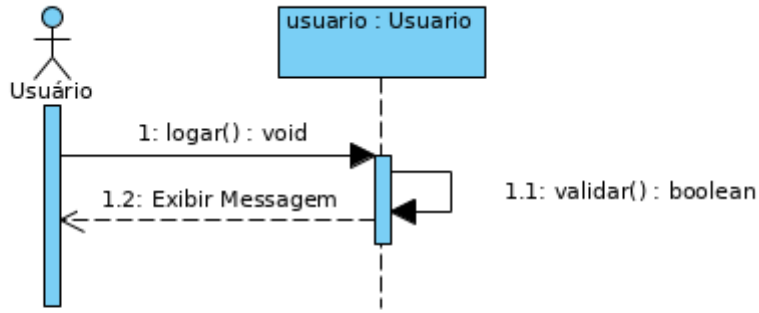

Figura 12: Login no sistema
Fonte: Os autores (2020)

O usuário loga no sistema e o mesmo válida os dados. O sistema retorna com uma mensagem de erro ou abre a página necessária.

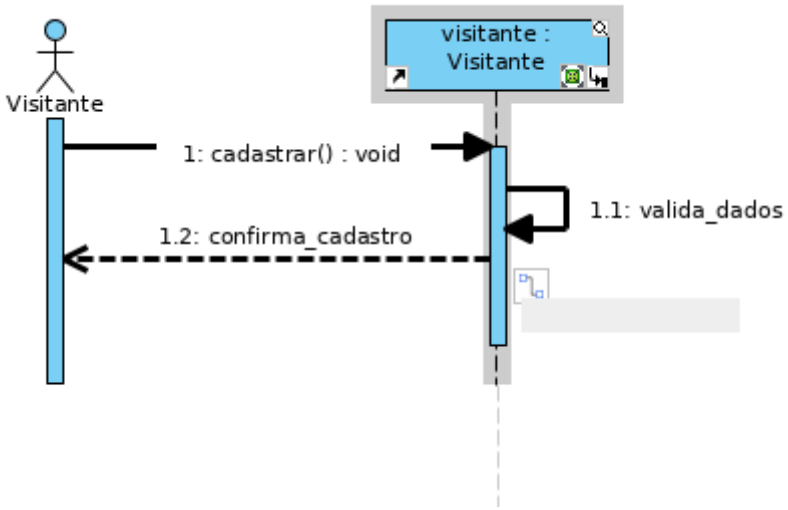

Figura 13: Cadastro de visitantes
Fonte: Os autores (2020)

O visitante acessa a área de cadastro e informa os dados necessário, pela função cadastrar. O sistema válida os dados e retorna uma mensagem confirmando esse cadastro.

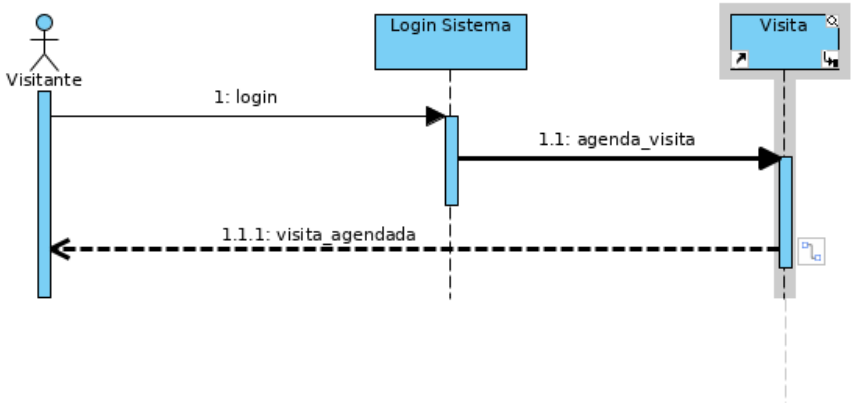

Figura 14: Agendamento de visita
Fonte: Os autores (2020)

O visitante, com seu usuário cadastrado, acessa o sistema e agenda uma visita ao museu, o sistema retorna com mensagem de visita agendada.

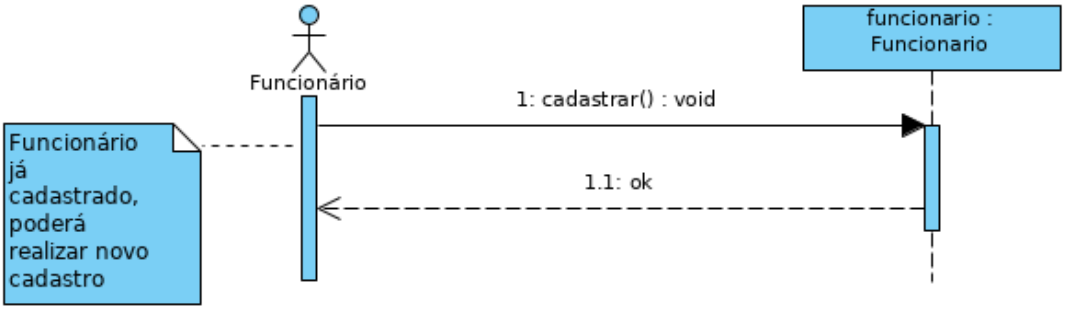

Figura 15: Cadastro de funcionário
Fonte: Os autores (2020)

O funcionário já cadastrado acessa o sistema e cadastra o novo funcionário, pela função cadastrar e retorna com uma mensagem de cadastro efetuado.

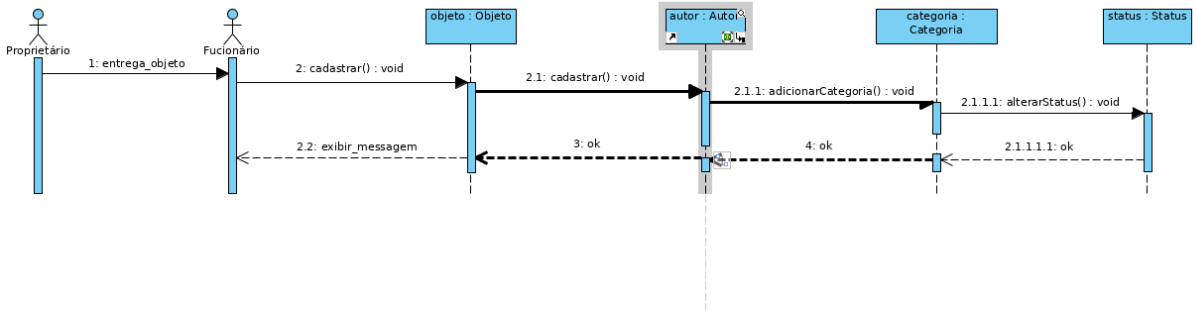

Figura 16: Cadastro de objeto
Fonte: Os autores (2020)

O proprietário entrega o objeto ao museu. O funcionário recebe esse objeto e faz o cadastro, cadastra o autor do objeto, adiciona a categoria e o status e o sistema retorna com mensagem informando desse cadastro realizado.

2.2 BANCO DE DADOS

2.2.1 Diagrama relacional

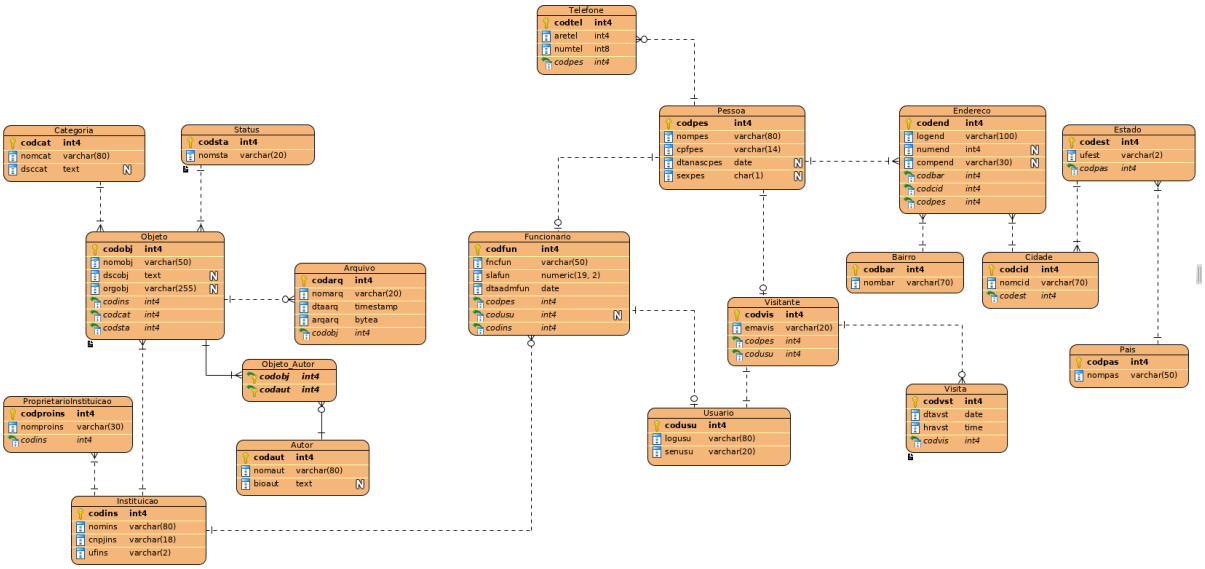

Figura 17: Diagrama relacional
Fonte: Os autores (2020)

O diagrama de entidade relacionamento possui várias tabelas, para cadastro de funcionários e visitantes até os objetos do acervo do museu.

A tabela Pessoa contem todas as informações principais para cadastrar um funcionário e visitante, como nome, CPF, data de nascimento e sexo. Interligada a ela tem a tabela Telefone, que armazena os telefones de cada um cadastrado, podendo ser opcional. Ainda interligada a tabela Pessoa, tem a tabela Endereco, onde armazena o logradouro, número e complemento.

Interligada a tabela Endereco, tem as tabelas Bairro, Cidade, Estado e Pais, que salvam apenas seus nomes.

As tabelas Funcionario e Visitante, estão conectadas a tabela Pessoa. Na tabela Funcionario, salva os dados como função, salário e data de admissão. Na tabela Visitante, fica apenas a informação de e-mail. Conectada a ela, tem a tabela de Visita que armazena a data e horário de uma visita ao museu.

Conectada as duas tabelas, tem a tabela Usuario, que serve para armazenar as informações de login e senha de cada usuário cadastrado ao museu.

A tabela Instituicao, salva as informações do museu, contendo o nome, CNPJ e o seu estado onde esta localizado. Conectado a ela, tem a tabela ProprietarioInstituicao, apenas para salvar o nome do proprietário. Essa tabela conecta a outras duas tabelas a de Funcionario,

armazenado dados dos mesmos, informando que eles estão conectados a instituição, e a Objeto, que armazena o acervo do museu.

A tabela Objeto contém as informações dos objetos, como nome, descrição e origem aproximada deste objeto. A ela se conecta as tabelas de Categoria e Status, onde a primeira contém as informações do nome e descrição e a segunda apenas seu status.

A tabela Arquivo armazena dados de arquivos relacionado ao objeto, como nome, data de inclusão e o arquivo em si. Essa tabela é opcional caso o objeto armazenado vai conter algum arquivo.

A tabela Autor é armazenado as informações de nome e biografia do mesmo relacionado ao objeto. Essa tabela poderá conter múltiplos autores de um objeto sendo necessário outra tabela para conectar ambas, nesse caso a tabela Objeto_Autor.

2.2.2 Arquivos de configuração e criação dos bancos

Para criar as tabelas para o projeto, foram utilizados os scripts abaixo. Com elas, as tabelas são criadas para a guardar os dados necessários para o sistema.

```
DROP TABLE IF EXISTS `pedido`;
CREATE TABLE `pedido` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `id_usuario` int(11) NOT NULL,
  `valor_total` decimal(10,0) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `id_usuario` (`id_usuario`),
  CONSTRAINT `pedido_ibfk_1` FOREIGN KEY (`id_usuario`) REFERENCES `usuario` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=9 DEFAULT CHARSET=latin1;

DROP TABLE IF EXISTS `pedido_item`;
CREATE TABLE `pedido_item` (
  `id_pedido` int(11) NOT NULL,
  `id_produto` int(11) NOT NULL,
  `valor` decimal(10,0) NOT NULL,
  UNIQUE KEY `idx_pedido_produto` (`id_pedido`,`id_produto`),
  KEY `id_produto` (`id_produto`),
  CONSTRAINT `pedido_item_ibfk_1` FOREIGN KEY (`id_pedido`) REFERENCES `pedido` (`id`),
  CONSTRAINT `pedido_item_ibfk_2` FOREIGN KEY (`id_pedido`) REFERENCES `pedido` (`id`),
  CONSTRAINT `pedido_item_ibfk_3` FOREIGN KEY (`id_produto`) REFERENCES `produto` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

DROP TABLE IF EXISTS `produto`;
CREATE TABLE `produto` (
  `id` int(10) NOT NULL AUTO_INCREMENT,
  `nome` varchar(200) NOT NULL,
  `valor` decimal(10,2),
  `imagem` varchar(1000) NOT NULL,
  `descricao` varchar(128) NOT NULL,
  `expor` char(1) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=40 DEFAULT CHARSET=latin1;

DROP TABLE IF EXISTS `usuario`;
CREATE TABLE `usuario` (
  `id` int(10) NOT NULL AUTO_INCREMENT,
  `nome` varchar(200) NOT NULL,
  `telefone` int(14) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=15 DEFAULT CHARSET=latin1;

CREATE TABLE `instituicao` (
  `id` int(4) NOT NULL AUTO_INCREMENT,
  `nome` varchar(40) NOT NULL,
  `matriz` char(1),
  `descricao` varchar(200),
  `imagem` varchar(1000) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `visita` (
  `id` int(10) NOT NULL AUTO_INCREMENT,
  `nome` varchar(60) NOT NULL,
  `fone` varchar(14) NOT NULL,
  `email` varchar(40) NOT NULL,
  `data` timestamp NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Quadro 2: Criação das tabelas
Fonte: Os autores (2020)

Para que o sistema tenha acesso ao banco de dados, foram utilizadas os seguintes scripts.

```
SELECT * FROM categoria;
SELECT * FROM instituicao ORDER BY id;
SELECT * FROM produto;
SELECT * FROM visita ORDER BY id desc;
```

Quadro 3: Selects de acesso
Fonte: Os autores (2020)

Os selects do quadro 3 são para acesso ao banco. Com tabelas mostrando a categoria, produto, a instituição ordenado pelo id e a vista, também ordenado pelo id mas por ordem decrescente.

```
SELECT * FROM instituicao WHERE id = . $_GET['id'];
SELECT * FROM produto WHERE id = . $_GET['id'];
```

Quadro 4: Selects de acesso conforme o id
Fonte: Os autores (2020)

Os selects acima selecionam a instituição e o objeto respectivamente.

```
UPDATE produto SET nome = ?, valor = ?, imagem = ?, descricao = ?, expor = ? WHERE id = ?;
DELETE FROM produto WHERE id = ?;
INSERT INTO produto (nome, valor, imagem, codcat) VALUES (?, ?, ?, ?);
INSERT INTO produto_visita (id_produto, data) VALUES (?, ?);
INSERT INTO visita (nome, fone, email, data) VALUES (?, ?, ?, ?);
```

Quadro 5: Insert, update e delete
Fonte: Os autores (2020)

O quadro 5 mostra a inserção de dados nas tabelas do banco com seus valores, além da opção de excluir o objeto do banco de dados. Outra opção é atualização de objeto no banco de dados, inserindo os respectivos valores.

```
SELECT p.* FROM produto p INNER JOIN categoria c ON p.codcat = c.codcat WHERE c.codcat = :categoria AND expor != 1 ORDER BY id desc;
SELECT COUNT(*) AS visita, produto.nome AS nome FROM produto_visita JOIN produto on produto.id = produto_visita.id_produto WHERE produto_visita.data = "' . date("Y-m-d") . "' GROUP BY produto_visita.id_produto ORDER BY visita desc;
SELECT c.*, COUNT(p.id) AS quantidade FROM categoria c LEFT JOIN produto p ON p.codcat = c.codcat GROUP BY c.codcat, c.nomcat ORDER BY quantidade;
```

Quadro 6: Scripts específicos de acesso ao banco
Fonte: Os autores (2020)

Os scripts do quadro 6, são para acesso específico ao banco de dados. O primeiro seleciona os objetos da tabela produto que contém uma determinada categoria e podem estar expostos.

O segundo select, conta o número de visitas que um objeto recebeu na página, exibindo o nome deste objeto.

O terceiro, exibe a categoria e a quantidade de objetos vinculados a ela, ordenando pela quantidade de objetos.

```
CREATE TABLE `historico`(
        `id` int(11) NOT NULL AUTO_INCREMENT,
        `horario` timestamp NOT NULL,
        `acao` varchar(10) NOT NULL,
        PRIMARY KEY (id)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Quadro 7: Tabela de auditoria
Fonte: Os autores (2020)

Para auditar o sistema, foi criado uma tabela para gravar o histórico com informações de horário e a ação realizada no banco de dados.

No quadro 8 tem os gatilhos criados para realizar as ações no banco de dados.

```
CREATE TRIGGER `tgr_objeto_update` AFTER UPDATE ON `produto`
FOR EACH ROW INSERT INTO historico(horario, acao) VALUES(NOW(), 'UPDATE');
CREATE TRIGGER `tgr_objeto_insert` AFTER INSERT ON produto
FOR EACH ROW INSERT INTO historico(horario, acao) VALUES(NOW(), 'INSERT');
CREATE TRIGGER `tgr_objeto_delete` AFTER DELETE ON produto
FOR EACH ROW INSERT INTO historico(horario, acao) VALUES(NOW(), 'DELETE');
```

Quadro 8: Trigger para a tabela auditoria
Fonte: Os autores (2020)

Cada um dos gatilhos será executado quando uma ação na tabela dos objetos for feita, verificando o que foi feito, a ação, e o horário do mesmo.

```
create database museu;
grant INSERT,SELECT,DELETE,UPDATE on museu.* to 'museu'@'%' identified by 'museu';
```

Quadro 9: Limitações de usuário
Fonte: Os autores (2020)

O script acima cria o banco de dados museu e limita o acesso ao usuário, onde ele poderá apenas fazer select, insert, delete e update nas tabelas do banco de dados.

2.3 PROGRAMAÇÃO

O site começa com a página de inicial, mostrando os objetos que estão no museu para exposição. O script a seguir demonstra como é criada a página inicial.

```
<?php
include('includes/header.php');

if(isset($_POST['categorias'])) {
   $categoria = $_POST['categorias'];
}
?>

<center>
   <br>
   <h1 class="mt-5">Exposição</h1>
</center>

<div class="col-md-3">
   <form action="." method="POST">
      <?php $queryCategoria = $conn->query("SELECT * FROM categoria"); ?>
      <select name="categorias" class="form-control">
         <option>-- Selecione uma categoria --</option>
         <?php while($row = $queryCategoria->fetch()) { ?>
               <option value="<?php echo $row['codcat']; ?>"><?php echo $row['nomcat'];
?></option>
         <?php } ?>
      </select>
      <button type="submit" class="btn btn-secondary">Listar</button>
   </form>
</div>

<div class="album py-5 bg-light">
   <div class="container">
      <div class="row">
         <?php
           $stmt = $conn->prepare("SELECT p.* FROM produto p INNER JOIN categoria c
ON p.codcat = c.codcat WHERE c.codcat = :categoria AND expor != 1 ORDER BY id
desc");
         $stmt->bindParam(":categoria", $categoria);
         $stmt->execute();
         while ($row = $stmt->fetch()) :
            ?>
            <div class="col-md-4">
               <div class="card mb-4 shadow-sm">
                  <?php if (!$row['imagem']) : ?>
                              <svg class="bd-placeholder-img card-img-top" width="100%"
height="225"     xmlns="http://www.w3.org/2000/svg"     preserveAspectRatio="xMidYMid
slice" focusable="false" role="img" aria-label="Placeholder: Thumbnail">
                        <title>Placeholder</title>
                        <rect width="100%" height="100%" fill="#55595c"/>
                          <text x="50%" y="50%" fill="#eceeef" dy=".3em"><?=$row['nome'] ?
></text>
                     </svg>
                  <?php else : ?>
```

```
                            <img src="./img/produtos/<?=$row['imagem']?>" width="100%"
height="225">
                    <?php endif; ?>

                    <div class="card-body">
                        <p class="card-text">
                            <?=$row['nome']?>
                        </p>

                        <div class="d-flex justify-content-between align-items-center">
                            <div class="btn-group">
                                    <form name="add" id="add" method="post" action="produto-
visualizar.php?id=<?=$row['id']?>">
                                <input type="hidden" name="id" value="<?=$row['id']?>"/>
                                        <button type="submit" class="btn btn-sm btn-outline-
secondary">Visualizar</button>
                                </form>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        <?php endwhile; ?>
    </div>
</div>

<?php
include('includes/footer.php');
?>
```

Quadro 10: Página index
Fonte: Os autores (2020)

O script contém dois arquivos, o arquivo footer.php e o arquivo header.php, mostrado rodapé e cabeçalho respectivamente. Os mesmos podem ser visto a seguir, começando pelo arquivo header.php.

```
<?php
include('connection.php');

function usuarioEstaLogado() {
    if (!isset($_SESSION['usuarioLogado']) || !$_SESSION['usuarioLogado']) {
        return false;
    }
    else {
        return true;
    }
}
?>
```

```
<!doctype html>
<html lang="pt-br">
<head>
   <meta charset="utf-8">
   <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
   <title>Museu da Computação  - UNOESC</title>
                  <link      href="https://getbootstrap.com/docs/4.4/dist/css/bootstrap.min.css"
rel="stylesheet">
   <link href="img/unoesc.ico" rel="icon">
</head>
<body>
   <header>
      <nav class="navbar navbar-expand-md navbar-dark fixed-top bg-dark">
            <a class="navbar-brand" href="index.php"><img width="55px" height="55px"
src="img/unoesc.png">Museu da Computação</a>
               <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarCollapse" aria-controls="navbarCollapse" aria-expanded="false" aria-
label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
         </button>
         <div class="collapse navbar-collapse" id="navbarCollapse">
            <ul class="navbar-nav mr-auto">
               <li class="nav-item">
                  <a class="nav-link" href="index.php">Home</a>
               </li>

               <?php if(usuarioEstaLogado()) {?>
                  <li class="nav-item">
                     <a class="nav-link" href="produto.php">Objeto</a>
                  </li>
               <?php }?>

               <li class="nav-item">
                  <a class="nav-link" href="instituicao.php">Instituição</a>
               </li>

               <li class="nav-item">
                  <a class="nav-link" href="visita.php">Visita</a>
               </li>

               <?php if(usuarioEstaLogado()) {?>
                  <li class="nav-item">
                     <a class="nav-link" href="relatorio.php">Relatório</a>
                  </li>
               <?php }?>

            </ul>
            <form class="form-inline mt-2 mt-md-0">
               <?php if (!usuarioEstaLogado()) : ?>
                                    <a  class="btn  btn-outline-success  my-2  my-sm-0"
```

```
href="login.php">Login</a>
                <?php endif; ?>

                <?php if (usuarioEstaLogado()) : ?>
                                        <a class="btn btn-outline-success my-2 my-sm-0"
href="logout.php">Sair</a>
                <?php endif; ?>
              </form>
            </div>
        </nav>
    </header>

    <main role="main">
<br>
<br>
```

Quadro 11: Arquivo header.php
Fonte: Os autores (2020)

O arquivo footer.php.

```
</main>

<script src="https://code.jquery.com/jquery-3.4.1.slim.min.js" integrity="sha384-
J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7imGFAV0wwj1yYfoRSJoZ+n"
crossorigin="anonymous"></script>
<script>window.jQuery || document.write('<script
src="https://getbootstrap.com/docs/4.4/assets/js/vendor/jquery.slim.min.js"><\/script>')</
script><script src="https://getbootstrap.com/docs/4.4/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
6khuMg9gaYr5AxOqhkVIODVIvm9ynTT5J4V1cfthmT+emCG6yVmEZsRHdxlotUnm"
crossorigin="anonymous"></script></body>

<div class="navbar fixed-bottom">
  <footer class="text-muted">
    <br><br>
    <div class="container">
      <p class="float-right">
        <a href="#">Voltar ao topo</a>
      </p>
    </div>
    <p>Copyright - Ciência da Computação UNOESC &amp;copy; Todos os direitos reservados.</
p>
  </footer>
</div>
</html>
```

Quadro 12: Arquivo footer.php
Fonte: Os autores (2020)

Para conectar ao banco, tem o arquivo conection.php.

```
<?php
ini_set('display_errors', 1);
```

```
ini_set('display_startup_errors', 1);
error_reporting(E_ALL);

$servername = "museu_db";
$username = "museu";
$password = "museu";
$db = "museu";

try {
   $conn = new PDO("mysql:host=$servername;dbname=$db", $username, $password);
   $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
   if (isset($_SESSION)) {
      session_destroy();
   }
   session_start();
} catch(PDOException $e) {
   echo "Connection failed: " . $e->getMessage();
}
?>
```

Quadro 13: Arquivo de conexão com o banco
Fonte: Os autores (2020)

O arquivo instituicao.php exibe os dados da instituição para qual é destinada.

```
<?php
   include('includes/header.php');
?>

<center>
   <br>
   <h1 class="mt-5">Nossas Instituições</h1>
</center>

<div class="album py-5 bg-light">
   <div class="container">
      <div class="row">
         <?php
         $stmt = $conn->query("SELECT * FROM instituicao ORDER BY id");
         while ($row = $stmt->fetch()) :
            ?>
            <div class="col-md-4">
               <div class="card mb-4 shadow-sm">
                  <?php if (!$row['imagem']) : ?>
                              <svg class="bd-placeholder-img card-img-top" width="100%"
height="225"     xmlns="http://www.w3.org/2000/svg"     preserveAspectRatio="xMidYMid
slice" focusable="false" role="img" aria-label="Placeholder: Thumbnail">

                        <title>Placeholder</title>
                        <rect width="100%" height="100%" fill="#55595c"/>
                         <text x="50%" y="50%" fill="#eceeef" dy=".3em"><?=$row['nome'] ?
```

```
></text>
                    </svg>
                <?php else : ?>
                        <img src="./img/instituicao/<?=$row['imagem']?>" width="100%"
height="225">
                <?php endif; ?>

                <div class="card-body">
                    <p class="card-text">
                        <?=$row['nome']?>
                    </p>
                    <div class="d-flex justify-content-between align-items-center">
                        <div class="btn-group">
                            <form name="add" id="add" method="post" action="instituicao-
visualizar.php?id=<?=$row['id']?>">
                                <input type="hidden" name="id" value="<?=$row['id']?>"/>
                                    <button type="submit" class="btn btn-sm btn-outline-
secondary">Detalhes</button>
                            </form>
                        </div>
                    </div>
                </div>
            </div>
        <?php endwhile; ?>
    </div>
  </div>
</div>

<?php
include('includes/footer.php');
?>
```

Quadro 14: Arquivo para a instituição
Fonte: Os autores (2020)

Para visualizar o a instituição, temos outro arquivo que mostra os dados da mesma.

```
<?php
include('includes/header.php');

$stmt = $conn->query("SELECT * FROM instituicao WHERE id = " .  $_GET['id']);
$row = $stmt->fetch();
?>

<h1 class="mt-5">Instituicao <?=$row['nome']?></h1>

<ul>
  <li>
    Descrição: <?=$row['descricao']?>
  </li>
  <li>
```

```
        Logo:
        <br>
        <img src="./img/instituicao/<?=$row['imagem']?>" width="300" height="200">
    </li>
</ul>

<a class="btn btn-secondary" href="index.php">Voltar</a>
<!--
    <a class="btn btn-primary" href="produto-alterar.php?id=<?=$row['id']?>">Alterar</a>
    <a class="btn btn-danger"
href="produto-remover.php?id=<?=$row['id']?>">Remover</a>
-->

<?php
    include('includes/footer.php');
?>
```

Quadro 15: Visualizar a instituição
Fonte: Os autores (2020)

Outros arquivos importantes são os arquivos de login e logout para que o usuário possa se conectar ao sistema.

```
<?php
include('includes/header.php');

if (isset($_POST['email']) && isset($_POST['senha'])) {
        if ($_POST['email'] == "admin@admin.com" && $_POST['senha'] == "admin") {
                $_SESSION['usuarioLogado'] = [
                        "email" => $_POST['email'],
                ];

                if (isset($_SESSION['url_redirect'])) {
                        $urlRedirect = $_SESSION['url_redirect'];
                        unset($_SESSION['url_redirect']);
                        //header('Location: ' . $_SESSION['url_redirect']);
                } else {
                        header('Location: index.php');
                }
        } else {
                header('Location: login.php');
        }
}
?>
<br>
<br>
<center>
<form class="form-signin" method="post">
        <h1 class="h3 mb-3 font-weight-normal">Faça login</h1>
        <label for="inputEmail" class="sr-only">Email address</label>
        <input type="email" id="inputEmail" name="email" class="form-control"
```

```
placeholder="Email" required autofocus style="width: 600px;">
        <br>
        <label for="inputPassword" class="sr-only">Password</label>
        <input type="password" id="inputPassword" name="senha" class="form-control"
placeholder="Senha" required style="width: 600px;">
        <div class="checkbox mb-3">
            <label>
                <input type="checkbox" value="remember-me"> Lembre-me
            </label>
        </div>
        <button class="btn btn-primary" type="submit">Logar</button>
</form>
<center>
<?php
include('includes/footer.php');
?>
```

Quadro 16: Login
Fonte: Os autores (2020)

```
<?php
include('includes/connection.php');

unset($_SESSION['usuarioLogado']);
header('Location: index.php');
?>
```

Quadro 17: Logout
Fonte: Os autores (2020)

O arquivo a seguir define a página para alterar o objeto que está em exposição no acervo do museu.

```
<?php
include('includes/header.php');

if (!$_SESSION['usuarioLogado']) {
   header('Location: login.php');
}
?>

<h1 class="mt-5">Objeto</h1>

<a class="btn btn-success" href="produto-novo.php">Novo Objeto</a>

<table class="table table-striped table-bordered table-hover">
   <thead>
      <tr>
         <th scope="col">Código</th>
         <th scope="col">Nome</th>
         <th scope="col">Valor</th>
         <th scope="col">Descrição</th>
         <th scope="col">Expor</th>
         <th scope="col">Ação</th>
      </tr>
   </thead>
```

```
  <tbody>
    <?php
    $stmt = $conn->query("SELECT * FROM produto");

    while ($row = $stmt->fetch()) {
      ?>
      <tr>
        <th scope="row">
          <?=$row['id']?>
        </th>
        <td>
          <?=$row['nome']?>
        </td>
        <td>
          <?=$row['valor']?>
        </td>
        <td>
          <?=$row['descricao']?>
        </td>
        <td>
          <?=$row['expor']?>
        </td>
        <td>
          <a href="produto-alterar.php?id=<?=$row['id']?>">Alterar</a>
          <a href="produto-remover.php?id=<?=$row['id']?>">Remover</a>
        </td>
      </tr>
      <?php
    }
    ?>
  </tbody>
</table>

<?php
include('includes/footer.php');
?>
```

Quadro 18: Visualizar o objeto para alteração
Fonte: Os autores (2020)

O objeto pode ser cadastrado, alterado ou removido. Os quadros a seguir mostram os scripts utilizados para fazer esse processo.

```
<?php
include('includes/header.php');

if (!usuarioEstaLogado()) {
  $_SESSION['url_redirect'] = 'produto-novo.php';
  header('Location: login.php');
}

if (isset($_POST['nome'])) {
  $image = null;
  if ($_FILES['imagem']) {
    $image = $_FILES['imagem']['name'];
    $file = './img/produtos/' . $image;
    move_uploaded_file($_FILES['imagem']['tmp_name'], $file);
```

```
      chmod($file , 0777);
   }

      $sqlUsuario = "INSERT INTO produto (nome, valor, imagem, codcat) VALUES (?, ?, ?, ?)";
   $stmtUsuario = $conn->prepare($sqlUsuario);
                  $stmtUsuario->execute([$_POST['nome'], $_POST['valor'], $image, $_POST['categoria']]);

   $id = $conn->lastInsertId();

   $conn->close;
   header('Location: produto-visualizar.php?id=' . $id);
}
?>

<h1 class="mt-5">Novo Objeto</h1>
<div class="album py-5 bg-light">
   <div class="container">
               <form name="novo-produto" id="novo-produto" method="post" action="" enctype="multipart/form-data">
         <div class="form-group">
            <label for="nome">Nome</label>
                <input type="text" class="form-control" id="nome" name="nome" required minlength="3" />
         </div>

         <div class="form-group">
            <label for="valor">Valor</label>
               <input type="number" class="form-control" id="valor" name="valor" required step="any" />
         </div>

         <div class="form-group">
            <label for="valor">Imagem</label>
            <br>
            <input type="file" id="imagem" name="imagem"/>
         </div>

         <div class="form-group">
            <label for="categoria">Categoria</label>
            <?php $queryCategoria = $conn->query("SELECT * FROM categoria"); ?>
            <select class="form-control" id="categoria">
               <?php while($row = $queryCategoria->fetch()) { ?>
                <option value="<?php echo $row['codcat']; ?>"><?php echo $row['nomcat']; ?></option>
               <?php } ?>
            </select>
         </div>
```

```
          <button type="submit" class="btn btn-primary">Salvar</button>
        </form>
    </div>
</div>

<?php
include('includes/footer.php');
?>
```

Quadro 19: Cadastro de objeto
Fonte: Os autores (2020)

```
<?php
include('includes/header.php');

if (!usuarioEstaLogado()) {
    $_SESSION['url_redirect'] = 'produto-novo.php';
    header('Location: login.php');
}


$stmt = $conn->query("SELECT * FROM produto WHERE id = " .  $_GET['id']);
$row = $stmt->fetch();

if (isset($_POST['id'])) {
    $image = null;
    if ($_FILES['imagem']) {
        $image = $_FILES['imagem']['name'];
        move_uploaded_file($_FILES['imagem']['tmp_name'], './img/produtos/' . $image);
    }

    $sqlUsuario = "UPDATE produto SET nome = ?, valor = ?, imagem = ?, descricao = ?, expor = ? WHERE id
= ?";
    $stmtUsuario = $conn->prepare($sqlUsuario);
    $stmtUsuario->execute([$_POST['nome'], $_POST['valor'], $image, $_POST['descricao'], $_POST['expor'],
$_POST['id']]);

    $conn->close;
    header('Location: produto-visualizar.php?id=' . $_POST['id']);
}
?>

<h1 class="mt-5">Alterar objeto <?=$row['id'] ?></h1>

<div class="album py-5 bg-light">
    <div class="container">
        <form name="novo-produto" id="novo-produto" method="post" action="" enctype="multipart/form-
data">
            <input type="hidden" id="id" name="id" value="<?=$row['id'] ?>">

            <div class="form-group">
                <label for="nome">Nome</label>
                <input type="text" class="form-control" id="nome" name="nome" required minlength="3"
value="<?=$row['nome'] ?>" />
            </div>

            <div class="form-group">
                <label for="valor">Valor</label>
                <input type="number" class="form-control" id="valor" name="valor" required step="any" value="<?
```

```
=$row['valor'] ?>" />
        </div>

        <div class="form-group">
            <label for="valor">Imagem</label>
            <br>
            <input type="file" id="imagem" name="imagem"/>
        </div>

        <div class="form-group">
            <label for="valor">Descrição</label>
            <input type="text" class="form-control" id="descricao" name="descricao" required step="any"
value="<?=$row['descricao'] ?>" />
        </div>

        <div class="form-group">
            <label for="valor">Expor</label>
            <input type="number" class="form-control" id="expor" name="expor" required step="any"
value="<?=$row['expor'] ?>" />
        </div>

        <div class="form-group">
            <label for="categoria">Categoria</label>
            <?php $queryCategoria = $conn->query("SELECT * FROM categoria"); ?>
            <select class="form-control" id="categoria">
                <?php while($row = $queryCategoria->fetch()) { ?>
                <option value="<?php echo $row['codcat']; ?>"><?php echo $row['nomcat']; ?></option>
                <?php } ?>
            </select>
        </div>

        <button type="submit" class="btn btn-primary">Salvar</button>
    </form>
  </div>
</div>

<?php
include('includes/footer.php');
?>
```

Quadro 20: Editar o objeto
Fonte: Os autores (2020)

```
<?php
include('includes/connection.php');

if ($_GET['id']) {
  try {
    $sql = "DELETE FROM produto WHERE id = ?";
    $stmt = $conn->prepare($sql);
    $stmt->execute([$_GET['id']]);
  } catch(Exception $e) {
    echo 'Produto não pode ser deletado';exit;
  }
}

header('Location: produto.php');
?>
```

Quadro 21: Remover objeto
Fonte: Os autores (2020)

Para visualizar a página é necessário outro arquivo para isso, o quadro 22 demonstra isso.

```
<?php
include('includes/header.php');

$stmt = $conn->query("SELECT * FROM produto WHERE id = " .  $_GET['id']);
$row = $stmt->fetch();

$sqlVisita = "INSERT INTO produto_visita (id_produto,data) VALUES (?, ?)";
$stmtVisita = $conn->prepare($sqlVisita);
$stmtVisita->execute([$_POST['id'], date("Y-m-d")]);
?>

<h1 class="mt-5">Produto <?=$row['id']?></h1>

<ul>
    <li>
        Código: <?=$row['id']?>
    </li>
    <li>
        Nome: <?=$row['nome']?>
    </li>
    <li>
        Descrição: <?=$row['descricao']?>
    </li>
    <li>
        Imagem:
        <br>
        <img src="./img/produtos/<?=$row['imagem']?>" width="200" height="200">
    </li>
</ul>

<a class="btn btn-secondary" href="index.php">Voltar</a>

<?php
    include('includes/footer.php');
?>
```

Quadro 22: Visualizar objeto
Fonte: Os autores (2020)

Os relatórios de visitas são demonstrados com o script a baixo.

```
<?php
    include('includes/header.php');
?>

<br><br>
<h2 class="container">Relação de agenda de visita</h2>
<div class="album py-5 bg-light">
    <div class="container">
        <div class="row">
            <?php
```

```
        $stmt = $conn->query("SELECT * FROM visita ORDER BY id desc");
        while ($row = $stmt->fetch()) :
            ?>
            <div class="col-md-4">
                <div class="card mb-4 shadow-sm">
                    <div class="card-body">
                        <p class="card-text">
                            <?=$row['nome']?><br>
                            <?=$row['fone']?><br>
                            <?=$row['email']?><br>
                            <?=$row['data']?>
                        </p>
                    </div>
                </div>
            </div>
        <?php endwhile; ?>
    </div>
  </div>
</div>


<h2 class="container">Relação de visualização ref produto</h2>
<div class="album py-5 bg-light">
  <div class="container">
    <div class="row">
      <?php
      $hoje = date("Y-m-d");
      $stmt = $conn->query("SELECT
      COUNT(*) AS visita, produto.nome as nome
    FROM
      produto_visita
      JOIN produto on produto.id = produto_visita.id_produto
    WHERE produto_visita.data = '" . date("Y-m-d") . "'
    GROUP BY produto_visita.id_produto
    ORDER BY visita desc;");
      while ($row = $stmt->fetch()) :
        ?>
        <div class="col-md-4">
          <div class="card mb-4 shadow-sm">
            <div class="card-body">
              <p class="card-text">
                <?=$row['nome']?>
                foi visualizado
                <?=$row['visita']?>
                <?php if ($row['visita'] > 1) {
                  echo 'vezes';
                } else {
                  echo 'vez';
                }
                ?>
              </p>
            </div>
          </div>
        </div>
      <?php endwhile; ?>
    </div>
  </div>
</div>
<?php
  include('includes/footer.php');
?>
```

Quadro 23: Relatório de visitas
Fonte: Os autores (2020)

Para agendar visitas é utilizado outro script, onde o visitante insere nome, telefone, e-mail e data para visita, fazendo inserção no banco.

```
<?php
   include('includes/header.php');
?>

<center>
   <h1 class="mt-5">Agendar Visita</h1>
   <p>Após agendar nós entraremos em contato.</p>

   <form method="post">
      <label>Nome: </label>
      <input type="text" name="nome" required style="width: 600px;"><br>

      <label>Fone: </label>
      <input type="tel" name="fone"  required style="width: 594px;"><br>


      <label>Email: </label>
      <input type="text" name="email" required style="width: 594px;"><br>

      <label>Data: </label>
      <input type="date"  name="dia" value="2020-06-10" min="2020-06-10" max="2020-
12-01"><br>

      <button class="btn btn-primary" name="enviar" type="submit">Agendar</button>
   </form>
</center>

<?php
   if(isset($_POST['enviar'])) {
      $sqlVisita = "INSERT INTO visita (nome ,fone, email, data) VALUES (?, ?, ?, ?)";
      $stmtVisita = $conn->prepare($sqlVisita);
      $stmtVisita->execute([$_POST['nome'], $_POST['fone'], $_POST['email'],
$_POST['dia']]);
      $idVisita = $conn->lastInsertId();
   }
?>

<?php
   include('includes/footer.php');
?>
```

Quadro 24: Registro de visitante
Fonte: Os autores (2020)

3 CONCLUSÃO

Algumas pessoas possuem o costume de guardar objetos para relembrar o passado e sua história. Outras, apenas guardam os objetos sem saber dar um destino correto ao mesmo. O presente trabalho tinha como objetivo criar um sistema para gerenciar o museu.

Levando em conta o projeto em si, foi feito de forma simples para facilitar a interação do usuário permitindo que o usuário possa ter uma boa interação com o sistema, facilitando a sua compreensão.