UNIVERSIDAD DE GUAYAQUIL

GRUPO A:

Cantos Miranda María Gabriela

Delgado Ordóñez Giancarlos Alejandro

Maldonado Maldonado Miguel Ángel

Tinoco Jiménez David Alejandro (líder)

Tema:
Tolerancia a Fallos

Carrera:
Software

Curso:
SOF–S–MA–6–3

Asignatura:
Construcción de Software

Docente:
Ph.D. Franklin Parrales Bravo

Descripción de Tolerancia a Fallos Aplicados

1. Mecanismos implementados

- Validación de campos requeridos en formularios (evita registros incompletos)

```php
private function registrarProveedor($postData) {
    $data = [
        'cedula' => isset($postData['cedula']) && $postData['cedula'] !== '' ? htmlspecialchars(trim($postData['cedul
        'nombre' => isset($postData['nombre']) ? htmlspecialchars(trim($postData['nombre'])) : '',
        'ruc' => isset($postData['ruc']) ? htmlspecialchars(trim($postData['ruc'])) : '',
        'telefono' => isset($postData['telefono']) && $postData['telefono'] !== '' ? htmlspecialchars(trim($postData[
        : null,
        'direccion' => isset($postData['direccion']) && $postData['direccion'] !== '' ? htmlspecialchars(trim($postDa
        ['direccion'])) : null,
        'correo' => isset($postData['correo']) && $postData['correo'] !== '' ? filter_var($postData['correo'],
        FILTER_SANITIZE_EMAIL) : null,
        'id_producto' => isset($postData['id_producto']) && $postData['id_producto'] !== '' ? (int)$postData['id_proc
    ];

    if (empty($data['nombre']) || empty($data['ruc']) || is_null($data['id_producto'])) {
        $this->setMensaje("Error: Nombre, RUC y Producto Asociado son campos obligatorios.", "error");
        return;
    }
    if (!empty($data['correo']) && !filter_var($data['correo'], FILTER_VALIDATE_EMAIL)) {
        $this->setMensaje("Error: El formato del correo electrónico no es válido.", "error");
        return;
    }
    if ($this->proveedorModelo->obtenerPorRuc($data['ruc'])) {
        $this->setMensaje("Error: Ya existe un proveedor con el RUC " . $data['ruc'] . ".", "error");
        return;
    }
```

- Validación de tipos de datos (números, fechas, etc.).

```php
private function registrarProveedor($postData) {
    $data = [
        'cedula' => isset($postData['cedula']) && $postData['cedula'] !== '' ? htmlspecialchars(trim($postData['cedula'])) : null,
        'nombre' => isset($postData['nombre']) ? htmlspecialchars(trim($postData['nombre'])) : '',
        'ruc' => isset($postData['ruc']) ? htmlspecialchars(trim($postData['ruc'])) : '',
        'telefono' => isset($postData['telefono']) && $postData['telefono'] !== '' ? htmlspecialchars(trim($postData['telefono']))
        : null,
        'direccion' => isset($postData['direccion']) && $postData['direccion'] !== '' ? htmlspecialchars(trim($postData
        ['direccion'])) : null,
        'correo' => isset($postData['correo']) && $postData['correo'] !== '' ? filter_var($postData['correo'],
        FILTER_SANITIZE_EMAIL) : null,
        'id_producto' => isset($postData['id_producto']) && $postData['id_producto'] !== '' ? (int)$postData['id_producto'] : null
    ];
```

- Manejo de errores con mensajes controlados en pantalla.

```
        $data = [
            $codigo,
            $_POST['nombre'],
            $_POST['descripcion'],
            $_POST['categoria'],
            $_POST['precio_venta'],
            $_POST['precio_compra'],
            $_POST['stock_inicial'],
            $_POST['stock_minimo']
        ];
        if ($this->productoModelo->insertar($data)) {
            $this->mensaje = "Producto registrado correctamente.";
            header('Location: ' . $_SERVER['PHP_SELF']);
            exit;
        } else {
            $this->mensaje = "Error al registrar el producto.";
        }
    }
```

- Comprobación de existencia antes de editar/eliminar productos.

```
private function registrar() {
    $codigo = $_POST['codigo'] ?? '';

    $productoExistente = $this->productoModelo->obtenerPorCodigo($codigo);
    if ($productoExistente) {
        $this->mensaje = "El código '$codigo' ya existe.";
        return;
    }
```

- Control de sesión para evitar acceso sin autenticación.

```php
public function __construct() {

    if (session_status() == PHP_SESSION_NONE) {
        session_start();
    }

    $this->proveedorModelo = new ProveedorModelo();
    $this->productoModelo = new ProductoModelo();

    // Inicializar datos para la vista
    $this->proveedores = $this->proveedorModelo->obtenerTodos();
    $this->productos = $this->productoModelo->obtenerTodos();

    // Manejar las acciones POST (agregar/actualizar/eliminar)
    if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        if (isset($_POST['accion'])) {
            $accion = $_POST['accion'];
            switch ($accion) {
                case 'registrar':
                    $this->registrarProveedor($_POST);
                    break;
                case 'actualizar':
                    $this->actualizarProveedor($_POST);
                    break;
                case 'eliminar':
                    $this->eliminarProveedor($_POST['ruc'] ?? '');
                    break;
            }
        }
    }
```

- Pruebas de desconexión de base de datos para ver la respuesta del sistema.

```php
<?php
class Conexion {
    public static function conectar() {
        $host = "localhost";
        $db = "inventario_db";
        $usuario = "root";
        $clave = "2004";

        try {
            $pdo = new PDO("mysql:host=$host;dbname=$db;charset=utf8", $usuario, $clave);
            $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
            return $pdo;
        } catch (PDOException $e) {
            die("Error de conexión: " . $e->getMessage());
        }
    }
}
```

- Eliminación lógica (estado activo/inactivo) en vez de borrado físico.

```php
public function eliminar($codigo) {
    $sql = "UPDATE productos SET estado = 0 WHERE codigo = ?";
    $stmt = $this->pdo->prepare($sql);
    return $stmt->execute([$codigo]);
}
```

```php
public function eliminar($id) {
    $sql = "UPDATE devoluciones SET estado = 0 WHERE id = ?";
    try {
        $stmt = $this->pdo->prepare($sql);
        return $stmt->execute([$id]);
    } catch (PDOException $e) {
        error_log("Error al eliminar devolución: " . $e->getMessage());
        return false;
    }
}
```

```php
public function eliminar($ruc) {
    $sql = "UPDATE proveedores SET estado = 0 WHERE ruc = ?";
    try {
        $stmt = $this->pdo->prepare($sql);
        return $stmt->execute([$ruc]);
    } catch (PDOException $e) {
        error_log("Error al eliminar proveedor: " . $e->getMessage());
        return false;
    }
}
```