PARCIAL 2 DE PRUEBAS DE SOFTWARE

Nombre: Aguirre Valdez Denilson Jefferson

Plataforma para vender motos

Este documento describe las pruebas de caja negra realizadas para una plataforma de venta de motos. Los códigos fuentes son los siguientes: “compra.js”, “moto.js”.

- Compra,js:

```
import Moto from './moto.js';

export default function realizarCompra(moto, direccionValida, pagoConfirmado) {
  if (!moto.estadoInventario) {
    return "Moto no disponible";
  }
  if (!direccionValida) {
    return "Dirección no válida";
  }
  if (!pagoConfirmado) {
    return "Pago no confirmado";
  }
  return "Compra completada";
}

```

- Moto.js

```
export default class Moto {
  constructor(precio, anio, estadoInventario = true) {
    this.precio = precio;
    this.anio = anio;
    this.estadoInventario = estadoInventario;
  }

  validarPrecio() {
    return this.precio >= 1000 && this.precio <= 20000;
  }

  validarAnio() {
    return this.anio >= 2000 && this.anio <= 2024;
  }
}

```

Las pruebas de caja negra incluyen partición de equivalencias, valores límites, transición de estado y tablas de decisiones.

1. Partición de equivalencias:

   En esta prueba se validan diferentes particiones de datos que representan rangos válidos e inválidos de precios y años de fabricación para las motos.

```
import { expect } from 'chai';
import Moto from '../moto.js';

describe('Pruebas de Partición de Equivalencias', () => {
  it('Debe validar precios dentro del rango permitido', () => {
    const motoValida = new Moto(15000, 2015);
    expect(motoValida.validarPrecio()).to.be.true;
  });

  it('Debe invalidar precios fuera del rango permitido', () => {
    const motoInvalida = new Moto(500, 2015);
    expect(motoInvalida.validarPrecio()).to.be.false;
  });

  it('Debe validar años dentro del rango permitido', () => {
    const motoValida = new Moto(15000, 2015);
    expect(motoValida.validarAnio()).to.be.true;
  });

  it('Debe invalidar años fuera del rango permitido', () => {
    const motoInvalida = new Moto(15000, 1995);
    expect(motoInvalida.validarAnio()).to.be.false;
  });
});

```

2. Valores limites

Se prueban los valores en los límites inferior y superior del rango permitido para precios y años de fabricación de las motos.

```
import { expect } from 'chai';
import Moto from '../moto.js';

describe('Pruebas de Valores Límites', () => {
  it('Debe validar precio mínimo', () => {
    const moto = new Moto(1000, 2015);
    expect(moto.validarPrecio()).to.be.true;
  });

  it('Debe invalidar precio justo fuera del límite inferior', () => {
    const moto = new Moto(999, 2015);
    expect(moto.validarPrecio()).to.be.false;
  });

  it('Debe validar precio máximo', () => {
    const moto = new Moto(20000, 2015);
    expect(moto.validarPrecio()).to.be.true;
  });

  it('Debe invalidar precio justo fuera del límite superior', () => {
    const moto = new Moto(20001, 2015);
    expect(moto.validarPrecio()).to.be.false;
  });

  it('Debe validar año mínimo', () => {
    const moto = new Moto(15000, 2000);
    expect(moto.validarAnio()).to.be.true;
  });

  it('Debe invalidar año justo fuera del límite inferior', () => {
    const moto = new Moto(15000, 1999);
    expect(moto.validarAnio()).to.be.false;
  });

  it('Debe validar año máximo', () => {
    const moto = new Moto(15000, 2024);
    expect(moto.validarAnio()).to.be.true;
  });

  it('Debe invalidar año justo fuera del límite superior', () => {
    const moto = new Moto(15000, 2025);
    expect(moto.validarAnio()).to.be.false;
  });
});
```

3. Transición de estados

Se prueban los diferentes estados posibles durante el proceso de compra de una moto y las transiciones válidas entre estos estados. Los estados que se consideran son:

- ✓ Compra completada
- ✓ Pago no confirmado
- ✓ Dirección no valida
- ✓ Moto no disponible

```
import { expect } from 'chai';
import Moto from '../moto.js';
import realizarCompra from '../compra.js';

describe('Pruebas de Transición de Estado', () => {
  it('Debe completar la compra con todas las condiciones válidas', () => {
    const moto = new Moto(15000, 2015);
    const resultado = realizarCompra(moto, true, true);
    expect(resultado).to.equal("Compra completada");
  });

  it('Debe fallar la compra si el pago no está confirmado', () => {
    const moto = new Moto(15000, 2015);
    const resultado = realizarCompra(moto, true, false);
    expect(resultado).to.equal("Pago no confirmado");
  });

  it('Debe fallar la compra si la dirección no es válida', () => {
    const moto = new Moto(15000, 2015);
    const resultado = realizarCompra(moto, false, true);
    expect(resultado).to.equal("Dirección no válida");
  });

  it('Debe fallar la compra si la moto no está en inventario', () => {
    const moto = new Moto(15000, 2015, false);
    const resultado = realizarCompra(moto, true, true);
    expect(resultado).to.equal("Moto no disponible");
  });
});
```

4. Tablas de decisiones

   Se validan diferentes decisiones que afectan el proceso de compra, como la confirmación del pago y la validez de la dirección de entrega.

```
import { expect } from 'chai';
import Moto from '../moto.js';
import realizarCompra from '../compra.js';

describe('Pruebas de Tablas de Decisiones', () => {
  it('Debe completar la compra con todas las condiciones válidas', () => {
    const moto = new Moto(15000, 2015);
    const resultado = realizarCompra(moto, true, true);
    expect(resultado).to.equal("Compra completada");
  });

  it('Debe fallar la compra si el pago no está confirmado', () => {
    const moto = new Moto(15000, 2015);
    const resultado = realizarCompra(moto, true, false);
    expect(resultado).to.equal("Pago no confirmado");
  });

  it('Debe fallar la compra si la dirección no es válida', () => {
    const moto = new Moto(15000, 2015);
    const resultado = realizarCompra(moto, false, true);
    expect(resultado).to.equal("Dirección no válida");
  });

  it('Debe fallar la compra si la moto no está en inventario', () => {
    const moto = new Moto(15000, 2015, false);
    const resultado = realizarCompra(moto, true, true);
    expect(resultado).to.equal("Moto no disponible");
  });
});
```