DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

Programación Web

NRC 2260

GABRIEL NICOLÁS VIVANCO RAZA

FECHA DE ENTREGA: 10/01/2025

TEMA:

"Análisis y Desarrollo: Clases en JavaScript para Sistemas Dinámicos"

Objetivo

Desarrollar un análisis profundo sobre la aplicabilidad y las mejores prácticas del uso de clases en JavaScript, y posteriormente implementar una solución funcional basada en los hallazgos para un problema real de software.

Instrucciones

1. Escenario

Estás trabajando como desarrollador en una empresa que busca crear una aplicación para administrar la operación de un servicio de transporte urbano inteligente. La plataforma debe gestionar registros de conductores, asignación de vehículos, control de rutas y monitoreo de tiempos de viaje.

Dado que la solución debe ser escalable, modular y fácil de mantener, la empresa ha decidido explorar la implementación de clases en JavaScript para estructurar el código de manera más efectiva.

2. Análisis

Analizar y responder a las siguientes preguntas para guiar tu desarrollo:

Abstracción y Modularidad: ¿Cómo estructurarías las clases para representar entidades como conductores, vehículos y rutas? ¿Qué propiedades y métodos serían necesarios para cada clase?

Tendríamos que basarnos en las siguientes clases:

Conductores: Información personal y vital como nombre, licencia, rutas. El método registrar ruta

ConductoresVIP: Tendría la herencia de la clase Conductores y la selección del vehículo. El método elegirVehiculo

Vehículos: Información vital de los vehículos como el modelo, placa, tipo de vehículo y método asignarVehiculo

Encapsulación: ¿Qué propiedades de cada clase deberían mantenerse privadas para proteger la integridad de los datos (por ejemplo, información personal del conductor)?

Las de información netamente personal como lo son licencia y placa

Herencia y Extensibilidad: ¿Cómo aprovecharías la herencia para diferenciar entre tipos de usuarios, como conductores regulares y conductores de servicios VIP?

La clase de ConductoresVIP va a heredar los atributos y métodos de la clase Conductores , y añadimos el nuevo método

Escalabilidad: ¿Qué prácticas implementarías para asegurar que el sistema pueda expandirse fácilmente conforme se añadan nuevas funcionalidades?

La modularidad y la herencia, para tener separación de las responsabilidades y también se tenga facilidad de poder añadir más clases y funcionalidades

3. Desarrollo

Implementa una solución funcional en JavaScript que cumpla con los siguientes criterios:

Definición de Clases:

Define una clase Conductor con propiedades básicas como nombre, licencia, y métodos como registrarRuta().

Implementa una clase Vehiculo para gestionar la asignación de vehículos, con propiedades como modelo y placa.

Herencia:

Extiende la clase Conductor para crear una clase ConductorVIP con beneficios adicionales como asignarVehiculoPreferido().

Encapsulación:

Utiliza propiedades privadas para proteger información sensible del conductor.

```
class Conductor{

    nombre;

    #licencia;



    constructor(nombre, licencia) {

        this.nombre = nombre;

        this.licencia = licencia;

        this.rutas = [];

    }



    registrarRuta(ruta) {
```

```
        this.rutas.push(ruta);
        console.log(`Ruta '${ruta}' asignada a ${this.nombre}.`);
    }

    obtenerInformacionConductor() {
        return {
            nombre: this.nombre,
            licencia: this.#licencia,
            rutas: this.rutas
        };
    }
}

class Vehiculo{
    modelo;
    #placa;
    tipoVehiculo;
    constructor(modelo, placa, tipoVehiculo){
        this.modelo = modelo;
        this.placa = placa;
        this.tipoVehiculo = tipoVehiculo;
        this.conductor = null;
    }

    asignarVehiculo(conductor){
```

```
        this.conductor = conductor;

        console.log(`Conductor asignado al vehículo ${this.modelo} con
placa: (${this.#placa})|| Tipo de vehiculo:.`);

    }

}



class ConductorVIP extends Conductor {

    constructor(nombre, licencia) {

        super(nombre, licencia);

        this.vehiculoElegido = null;

    }



    asignarVehiculoElegido(vehiculo) {

        this.vehiculoElegido = vehiculo;

        console.log(`Vehículo eleigod por el conductor VIP:
${vehiculo.modelo} (${vehiculo.placa}).`);

    }



    registrarRuta(ruta) {

        super.registrarRuta(ruta);

        console.log(`Ruta registrada del conducto VIP ->
${this.nombre}.`);

    }

}



const conductor1 = new Conductor("Andre Limaico", "X-171427-X");
```

```
const vehiculo1 = new Vehiculo("Mazda CX3", "PQD-488", "SUV");

const conductorVIP1 = new ConductorVIP("jose Lizarzaburu",
"X-142890-X");



conductor1.registrarRuta("Ruta Alborada");

vehiculo1.asignarVehiculo(conductor1);



conductorVIP1.registrarRuta("Ruta La Carolina VIP");

conductorVIP1.asignarVehiculoElegido(vehiculo1);
```