Resolución de la etapa 2

Realizar la clase `Cliente` *en JavaScript con la ayuda de ChatGPT.*

Clase *Cliente*

Atributos:

- Nombre.
- Apellido.
- DNI.
- Correo electrónico.
- Contraseña.

Métodos:

- Constructor para inicializar los atributos.
- *Getters* y *setters* para acceder y modificar los atributos de manera controlada.
- Métodos adicionales del comportamiento.

La clase **`Cliente`** encapsula los **datos** personales y de autenticación del usuario.

Los ***getters*** y ***setters*** permiten acceder y modificar los **atributos** de manera controlada y segura.

A este ejemplo, se le agregarán **validaciones en el constructor y en los *setters*** para mantener la consistencia de los objetos.

Se puede iterar la clase con el *Large Language Models (LLM)* y mejorarla.

```javascript
class Cliente {
    constructor(nombre, apellido, dni, email, password) {
        this.nombre = nombre;
        this.apellido = apellido;
        this.dni = dni;
        this.email = email;
        this.password = password;
    }

    // Getters y setters
    getNombre() {
        return this.nombre;
    }

    setNombre(nombre) {
        this.nombre = nombre;
    }

    // Métodos adicionales según requerimientos
}
```

```
class Cliente {
    // Atributos privados de la clase
    #password;

    constructor(id, nombre, email, password, confirmPassword) {
        if (!this.validarNombre(nombre)) throw new Error('Nombre inválido');
        if (!this.validarEmail(email)) throw new Error('Email inválido');
        if (!this.validarPassword(password)) throw new Error('Password inválido');
        if (password !== confirmPassword) throw new Error('Las contraseñas no coinciden');

        this._id = id; // Identificador único
        this._nombre = nombre;
        this._email = email;
        this.#password = password;
    }
```

...

```
...
    // Validar el nombre
    validarNombre(nombre) {
        return typeof nombre === 'string' && nombre.trim().length > 0;
    }

    // Validar el correo electrónico
    validarEmail(email) {
        const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
        return emailRegex.test(email);
    }

    // Validar la contraseña
    validarPassword(password) {
        return typeof password === 'string' && password.length >= 6;
    }
...
```

```
...
    // Getter para el nombre
    get nombre() {
        return this._nombre;
    }

    // Getter para el email
    get email() {
        return this._email;
    }

    get password() {
        return this.#password;
    }
...
```

```
...
    // Setter para el nombre
    set nombre(nuevoNombre) {
        if (!this.validarNombre(nuevoNombre)) throw new Error('Nombre inválido');
        this._nombre = nuevoNombre;
    }

    // Setter para el email
    set email(nuevoEmail) {
        if (!this.validarEmail(nuevoEmail)) throw new Error('Email inválido');
        this._email = nuevoEmail;
    }
...
```

```
    // Método para actualizar la contraseña
    actualizarPassword(nuevaPassword, confirmPassword) {
        if (!this.validarPassword(nuevaPassword)) throw new Error('Password inválido');
        if (nuevaPassword !== confirmPassword) throw new Error('Las contraseñas no coinciden');
        this.#password = nuevaPassword;
    }
}
```

Pruebas unitarias

Se realizan pruebas para verificar que la **creación y modificación de clientes** funcionen correctamente.

Se utiliza el *framework* ***Jest*** para escribir y ejecutar las pruebas unitarias.

javascript — Copiar código

```javascript
// Ejemplo de pruebas unitarias para la clase Cliente
describe('Cliente', () => {
    it('Debería crear un nuevo cliente correctamente', () => {
        const cliente = new Cliente('Juan', 'Perez', '12345678', 'juan@example.com
        expect(cliente.getNombre()).toBe('Juan');
        // Asegurar otros atributos
    });

    it('Debería permitir modificar el nombre del cliente', () => {
        const cliente = new Cliente('Juan', 'Perez', '12345678', 'juan@example.com
        cliente.setNombre('Pedro');
        expect(cliente.getNombre()).toBe('Pedro');
    });

    // Más pruebas según requerimientos
});
```