Universidad de las Fuerzas Armadas ESPE

Departamento de Ciencias de la Computación

Programación Integrativa de Componentes Web

Nombre: Vinicio Guaman

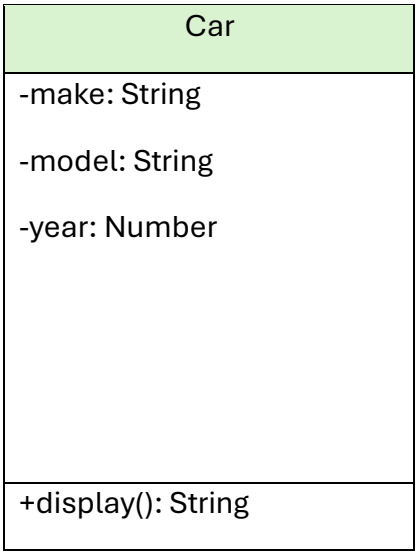
NRC: 16496

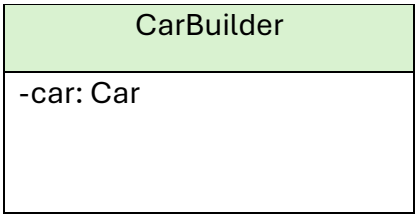
Fecha de entrega: 25/06/24

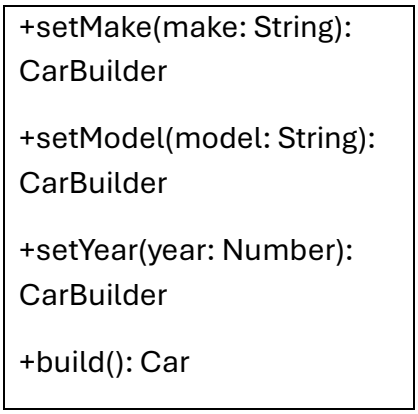
Tarea 1.6: pattern design with javascript

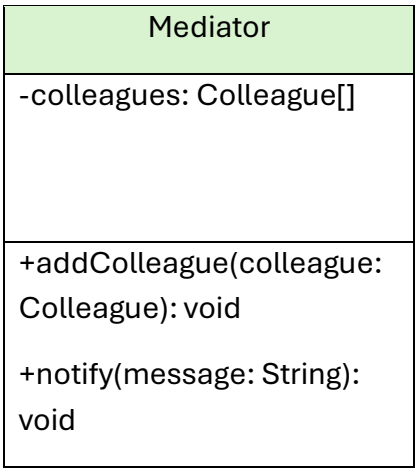
GENERAR UNA APLICACION con javascript basada en patrones de diseño y modelo de clases asociados con java y javacript desde un repositorio con ejemplos resueltos. Generar la aplicación web y documentar incluyendo diagramas de clases, codigo fuente y publicar en portal web de la espe y SERVIDOR FTP (servicios-integrados.net)

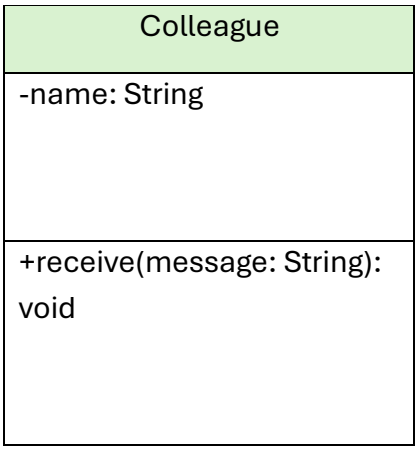
Diagramas de Clases






+setMake(make: String): CarBuilder
+setModel(model: String): CarBuilder
+setYear(year: Number): CarBuilder
+build(): Car


Mediator
-colleagues: Colleague[]
+addColleague(colleague: Colleague): void
+notify(message: String): void


Colleague
-name: String
+receive(message: String): void

Código Utilizado

HTML

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Sistema de Mensajería</title>
    <link rel="stylesheet" href="CSS/style.css">
</head>
<body>
    <div class="container">
        <h1>Sistema de Mensajería</h1>
        <button id="createMessageBtn">Crear Mensaje</button>
        <button id="sendMessageBtn">Enviar Mensaje</button>
        <div id="output"></div>
    </div>
    <script src="JS/script.js"></script>
</body>
</html>
```

CSS

```css
body {
    font-family: Arial, sans-serif;
    background-color: #f0f0f0;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
    margin: 0;
}

.container {
    background-color: white;
    padding: 20px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    border-radius: 5px;
    text-align: center;
}

button {
```

```
    margin: 10px;
    padding: 10px 20px;
    font-size: 16px;
    cursor: pointer;
}

#output {
    margin-top: 20px;
}
```

JS

```
// Builder Pattern
class Message {
    constructor() {
        this.sender = '';
        this.recipient = '';
        this.content = '';
    }

    display() {
        return `Mensaje de ${this.sender} a ${this.recipient}: ${this.content}`;
    }
}

class MessageBuilder {
    constructor() {
        this.message = new Message();
    }

    setSender(sender) {
        this.message.sender = sender;
        return this;
    }

    setRecipient(recipient) {
        this.message.recipient = recipient;
        return this;
    }

    setContent(content) {
        this.message.content = content;
        return this;
```

```
    }

    build() {
        return this.message;
    }
}

// Mediator Pattern
class ChatRoom {
    constructor() {
        this.users = [];
    }

    addUser(user) {
        this.users.push(user);
    }

    sendMessage(message) {
        this.users.forEach(user => user.receive(message));
    }
}

class User {
    constructor(name) {
        this.name = name;
    }

    receive(message) {
        const output = document.getElementById('output');
        const p = document.createElement('p');
        p.textContent = `${this.name} recibió: ${message}`;
        output.appendChild(p);
    }
}

// Funcionalidad de la Aplicación
document.getElementById('createMessageBtn').addEventListener('click', () => {
    const builder = new MessageBuilder();
    const message =
builder.setSender('Vinicio').setRecipient('Paul').setContent('Hola, ¿cómo
estás?').build();
    const output = document.getElementById('output');
    const p = document.createElement('p');
    p.textContent = message.display();
    output.appendChild(p);
```

```
});

document.getElementById('sendMessageBtn').addEventListener('click', () => {
    const chatRoom = new ChatRoom();
    const user1 = new User('Vinicio');
    const user2 = new User('Paul');
    chatRoom.addUser(user1);
    chatRoom.addUser(user2);
    chatRoom.sendMessage('Hola a todos desde el mediador');
});
```