Informe de Laboratorio 04

Tema: Ajax

Nota

Estudiante	Escuela	Asignatura
Luis Guillermo Luque Condori, Fernando Miguel Garambel Marín, William Herderson Choquehuanca Berna, Jeans Anthony Ajra Huacso lluquecon@unsa.edu.pe fgarambel@unsa.edu.pe ajrahuacso@unsa.edu.pe wchoquehuancab@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Laboratorio de P web Semestre: III Código: 20233478

Laboratorio	Tema	Duración
04	Ajax	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - A	Del 4 de mayo55555	Al 8 de mayo 2024

1. Tarea Documento

- Listar los archivos Markdown disponibles
- Ver el contenido de un archivo Markdown traducido a HTML
- Crear nuevos archivos MarkDown y almacenarlos en el servidor

2. Tarea Plataforma

- Liste todas las "regiones"
- Muestre el número total de confirmados por región
- Encuentre las 10 regiones cuya suma total sea la mayor
- Visualice un gráfico en el tiempo de los valores para la región de Arequipa
- Haga gráficos comparativos entre regiones usando líneas
- Visualice un gráfico comparativo del crecimiento en regiones excepto Lima y Callao
- Haga gráficos comparativos entre regiones elegidas por el usuario.
- Visualice un gráfico comparativo del crecimiento en regiones excepto Lima y Callao, mostrando el número de confirmados por cada día

3. URL GitHub

- `https://github.com/FernandoGarambelM/Ajax.git`

4. Tareas Documento

4.1. Listar los archivos Markdown disponibles

4.2. Ver el contenido de un archivo Markdown traducido a HTML

4.3. Crear nuevos archivos MarkDown y almacenarlos en el servidor

5. Tareas Plataforma

5.1. Liste las regiones

- HTML

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="style.css">
    <title>Ejercicio1</title>
</head>
<body>
    <div id="datos">
        <h1>LISTA DE REGIONES</h1>
        <button id="btnMostrarRegiones">Mostrar Regiones</button>
        <div class="contenedor">
            <ul id="regionList"></ul>
        </div>
    </div>
    <script src="ejercicio1.js"></script>
</body>
</html>
```

- Script

```javascript
document.addEventListener("DOMContentLoaded", () => {
    btnMostrarRegiones.addEventListener('click', () => {
    });
});

function mostrarRegiones(data) {
    const regionList = document.getElementById('regionList');
    data.forEach(regionData => {
        const li = document.createElement('li');
        li.textContent = regionData.region;
        regionList.appendChild(li);
    });
}
```

- Página

- Resultado

5.2. Muestre el número total de confirmados por región

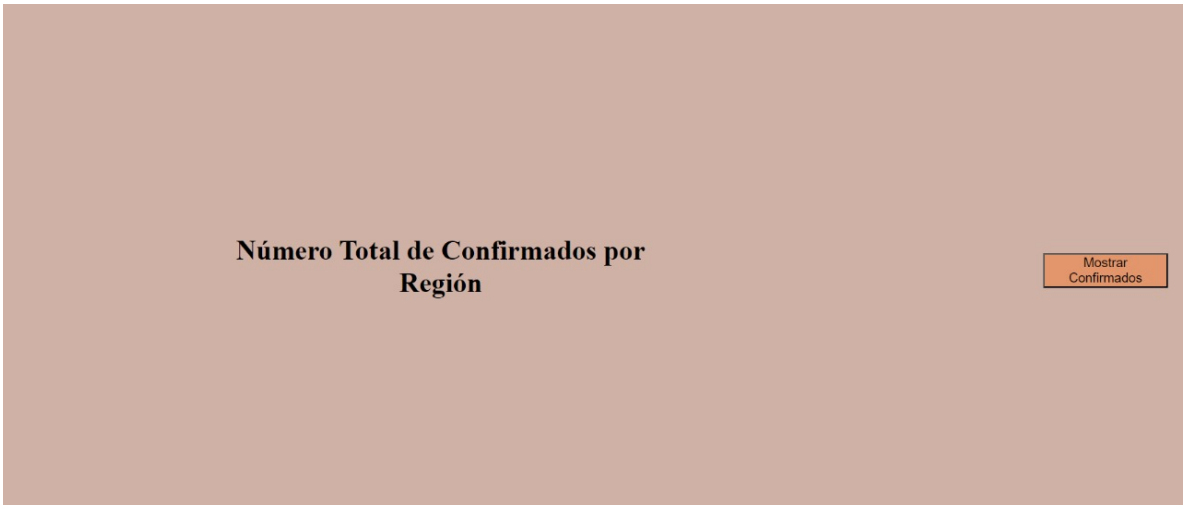
- HTML

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Confirmados por Región</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <h1>Número Total de Confirmados por Región</h1>
    <div id="regiones"></div>
    <button id="btnConfirmadosRegion">Mostrar Confirmados</button>

    <script src="ejercicio2.js"></script>
</body>
</html>
```

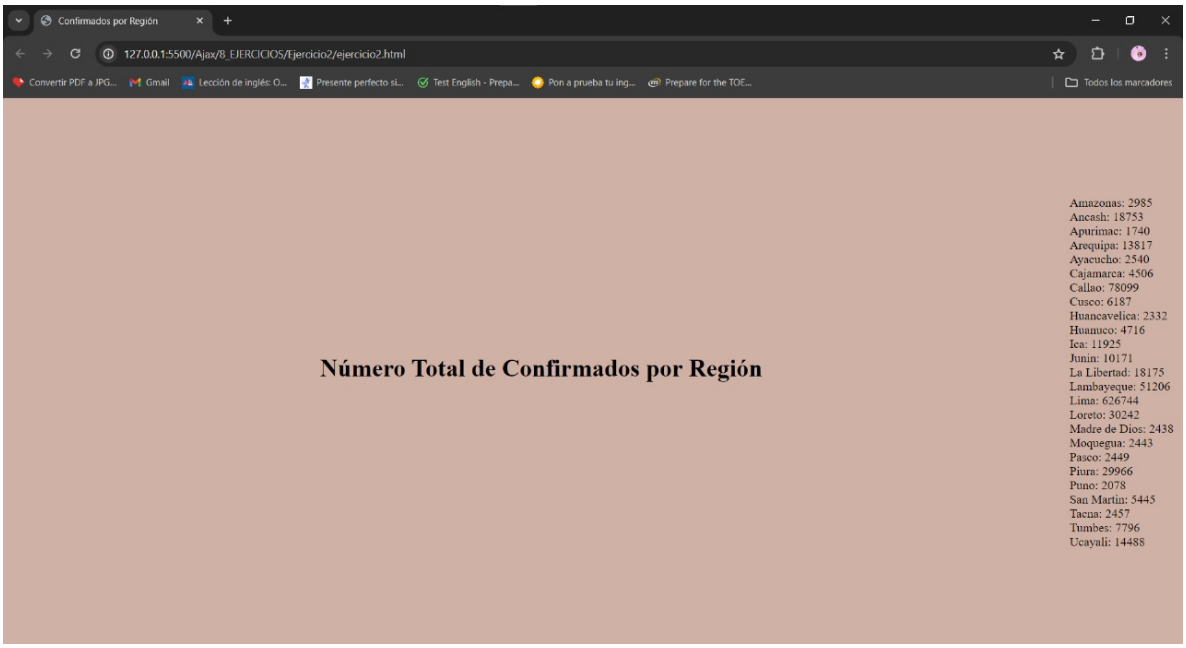
- Script

```javascript
document.addEventListener("DOMContentLoaded", function() {
    const btnConfirmadosRegion = document.getElementById('btnConfirmadosRegion');

    btnConfirmadosRegion.addEventListener('click', () => {
        // Realizar la solicitud
        let xhr = new XMLHttpRequest();
        xhr.open("GET", "../data.json", true);
        xhr.onload = function() {
            if (xhr.status == 200) {
                var data = JSON.parse(xhr.responseText);
                mostrarConfirmadosPorRegion(data);
                btnConfirmadosRegion.style.display = 'none';
            } else {
                console.error('Error en la red:', xhr.statusText);
            }
        };
        xhr.onerror = function() {
            console.error('Error en la red.');
        };
        xhr.send();
    })

});

function mostrarConfirmadosPorRegion(data) {
    const regionesDiv = document.getElementById("regiones");

    // Objeto para almacenar el total de confirmados por region
    const confirmadosPorRegion = {};

    // Calcular el total de confirmados por región
    data.forEach((regionData) => {
        const region = regionData.region;
        const confirmados = regionData.confirmed.reduce((total, confirmado) => {
            return total + parseInt(confirmado.value);
        }, 0);
        confirmadosPorRegion[region] = confirmadosPorRegion[region] ? confirmadosPorRegion[region] + confirma
    });

    // Mostrar el total por región
    for (let region in confirmadosPorRegion) {
        const totalConfirmados = confirmadosPorRegion[region];
        const paragraph = document.createElement("p");
        paragraph.textContent = region + ": " + totalConfirmados;
        regionesDiv.appendChild(paragraph);
    }
```

- Página



- Resultado



5.3. Encuentre las 10 regiones cuya suma total sea la mayor

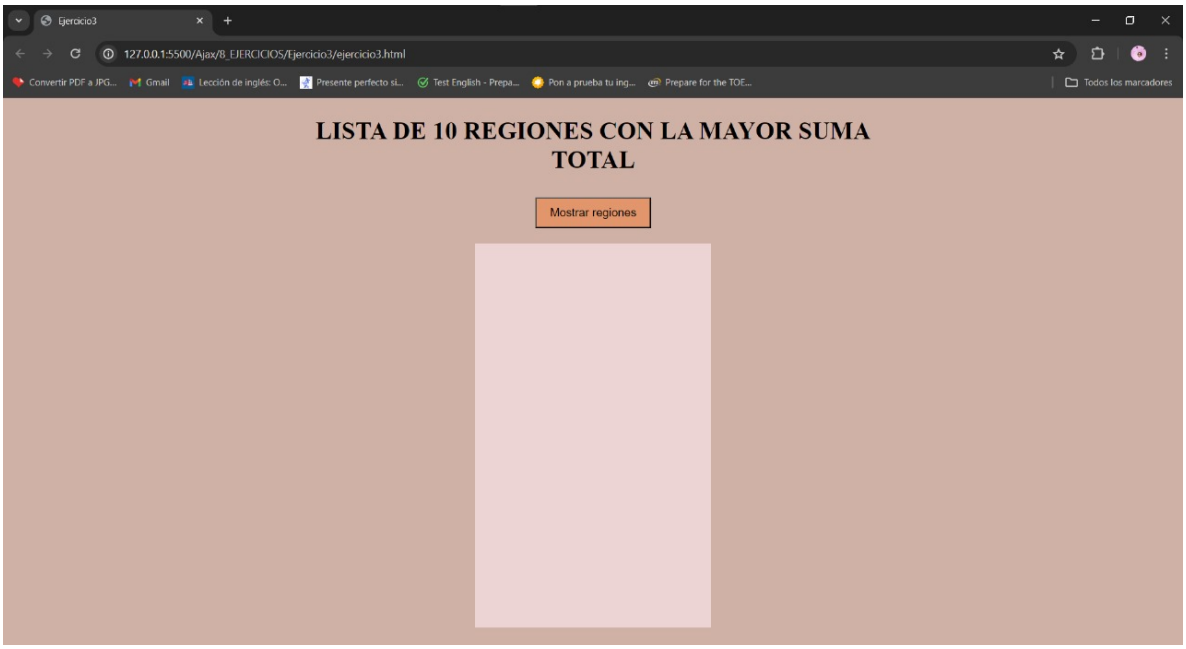
- HTML

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="style.css">
    <title>Ejercicio3</title>
</head>
<body>
    <div id="datos">
        <h1>LISTA DE 10 REGIONES CON LA MAYOR SUMA TOTAL</h1>
        <button id="btnMostrarRegiones">Mostrar regiones</button>
        <div class="contenedor">
            <ul id="regionList"></ul>
        </div>
    </div>
    <script src="ejercicio3.js"></script>
</body>
</html>
```

- Script

```javascript
document.addEventListener("DOMContentLoaded", () => {
    const btnMostrarRegiones = document.getElementById('btnMostrarRegiones');

    btnMostrarRegiones.addEventListener('click', () => {
        fetch('../data.json')
            .then(response => {
                if (!response.ok) {
                    throw new Error('Error en la red');
                }
                return response.json();
            })
            .then(data => {
                mostrarRegiones(data);
                btnMostrarRegiones.style.display = 'none';
            })
            .catch(error => {
                console.error('Error:', error);
            });
    });
});

function mostrarRegiones(data) {
    const regionList = document.getElementById('regionList');
    const casosConfirmadosPorRegion = {};

    data.forEach(regionData => {
        const region = regionData.region;
        let totalCasosConfirmados = 0;
        regionData.confirmed.forEach(caso => {
            totalCasosConfirmados += parseInt(caso.value);
        });
        casosConfirmadosPorRegion[region] = totalCasosConfirmados;
    });

    while(Object.keys(casosConfirmadosPorRegion).length > 10){
        let menorCantidad = Infinity;
        let regionMenorCasos = null;
        for (let region in casosConfirmadosPorRegion) {
            const cantidadCasos = casosConfirmadosPorRegion[region];
            if (cantidadCasos < menorCantidad) {
                menorCantidad = cantidadCasos;
                regionMenorCasos = region;
            }
        }
        delete casosConfirmadosPorRegion[regionMenorCasos];
    }

    for (let region in casosConfirmadosPorRegion) {
        const li = document.createElement('li');
        li.textContent = `${region}: ${casosConfirmadosPorRegion[region]}`;
        regionList.appendChild(li);
    }

    const totalLi = document.createElement('li');
    totalLi.style.marginTop = '20px';
    totalLi.textContent = `Total de casos: ${contarTotalCasos(casosConfirmadosPorRegion)}`;
    regionList.appendChild(totalLi);
}

function contarTotalCasos(casosConfirmadosPorRegion) {
    let totalCasos = 0;
    for (let region in casosConfirmadosPorRegion) {
        const casosRegion = casosConfirmadosPorRegion[region];
        totalCasos += casosRegion;
    }
    return totalCasos;
}
```

- Página



- Resultado

5.4. Visualice un gráfico en el tiempo de los valores para la región de Arequipa

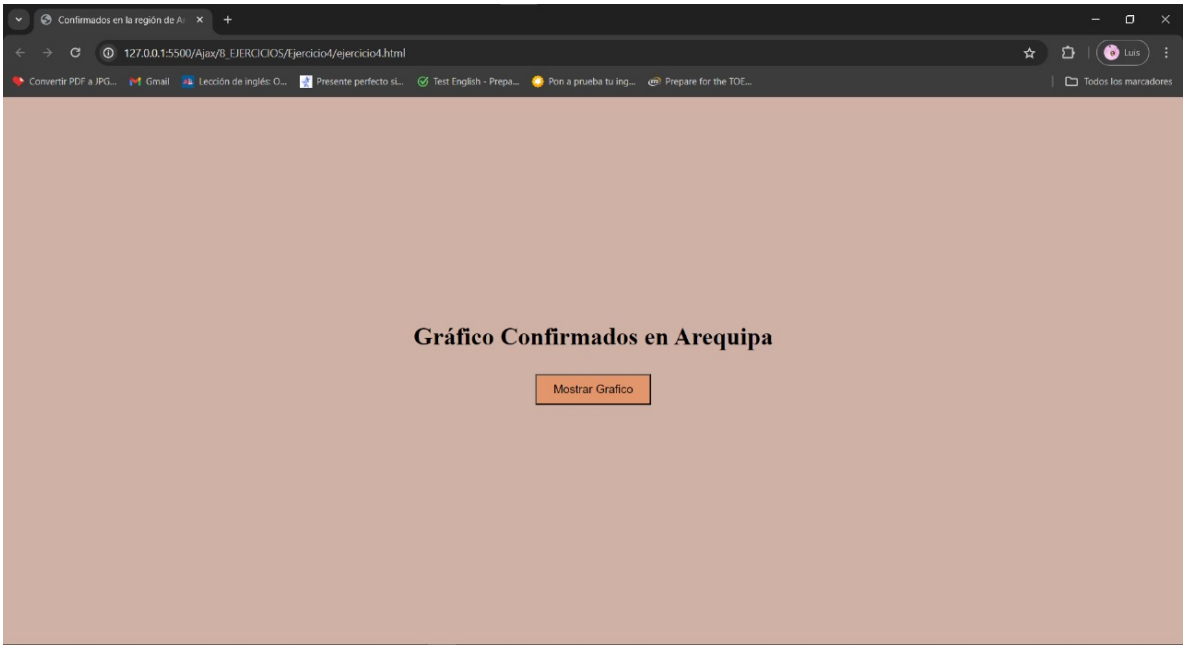
- HTML

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Confirmados en la región de Arequipa</title>
    <link rel="stylesheet" href="style.css">

</head>
<body>
    <div id="datos">
        <h1>Gráfico Confirmados en Arequipa</h1>
        <button id="mostrarGrafico">Mostrar Grafico</button>
        <div class="contenedor">
            <canvas id="Grafico" width="400" height="200"></canvas>
        </div>

        <script src="ejercicio4.js"></script>
        <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    </div>
</body>
</html>
```

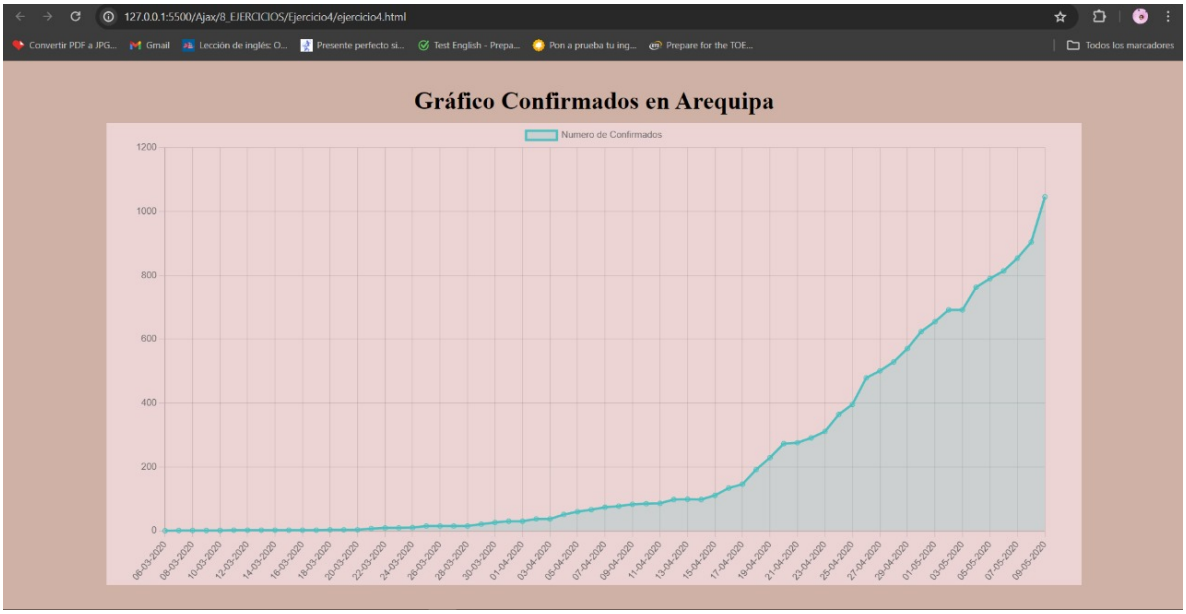
- Script

```javascript
document.addEventListener("DOMContentLoaded", function(){
    const btnMostrarGrafico = document.getElementById('mostrarGrafico');
    const contenedor = document.querySelector('.contenedor');

    btnMostrarGrafico.addEventListener('click', () => {
        let xhr = new XMLHttpRequest();
        xhr.open("GET", "../data.json", true);
        xhr.onload = function() {
            if (xhr.status === 200) {
                let data = JSON.parse(xhr.responseText);
                let arequipaData = data.find(region => region.region === "Arequipa");
                if (arequipaData) {
                    mostrarGrafico(arequipaData.confirmed);
                    contenedor.style.display = 'flex';
                    btnMostrarGrafico.style.display = 'none';
                } else {
                    console.error("No se econtraron datos de Arequipa");
                }
            } else {
                console.error('Error en la red:',xhr.statusText);
            }
        };
        xhr.onerror = function() {
            console.error('Error en la red');
        };
        xhr.send();
    });

})

function mostrarGrafico(arequipaData) {
    let dates = arequipaData.map(entry => entry.date);
    let values = arequipaData.map(entry => parseInt(entry.value));

    let ctx = document.getElementById('Grafico').getContext('2d');
    let graphic = new Chart(ctx, {
        type: 'line',
        data: {
            labels: dates,
            datasets: [{
                label: 'Numero de Confirmados',
                data: values,
                borderColor: 'rgba(75, 192, 192, 1)',
                backgroundColor: 'rgba(75, 192, 192, 0.2)',
                fill: true,
            }]
```

- Página



- Resultado



5.5. Haga gráficos comparativos entre regiones usando líneas

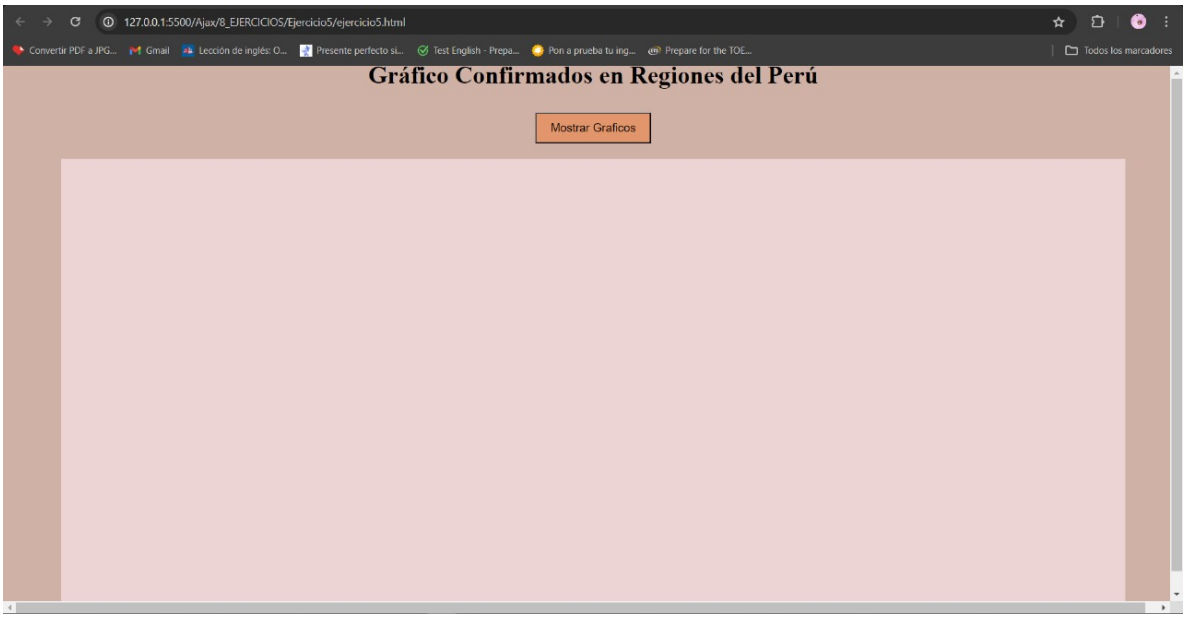
- HTML

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejercicio5</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <div id="datos">
        <h1>Gráfico Confirmados en Regiones del Perú</h1>
        <button id="mostrarGrafico">Mostrar Graficos</button>
        <div class="contenedor"></div>
    </div>

    <script src="ejercicio5.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</body>
</html>
```

- Script

```
document.addEventListener("DOMContentLoaded", function() {
    const btnMostrarGrafico = document.getElementById('mostrarGrafico');
    const contenedor = document.querySelector('.contenedor');

    btnMostrarGrafico.addEventListener('click', () => {
        let xhr = new XMLHttpRequest();
        xhr.open("GET", "../data.json", true);
        xhr.onload = function() {
            if (xhr.status === 200) {
                let data = JSON.parse(xhr.responseText);
                if (data.length > 0) {
                    mostrarGraficos(data);
                    contenedor.style.display = 'flex';
                    btnMostrarGrafico.style.display = 'none';
                } else {
                    console.error("No se encontraron datos de regiones");
                }
            } else {
                console.error('Error en la red:', xhr.statusText);
            }
        };
        xhr.onerror = function() {
            console.error('Error en la red');
        };
        xhr.send();
    });
});

function mostrarGraficos(data) {
    const contenedor = document.querySelector('.contenedor');
    contenedor.innerHTML = ''; // Clear any existing content

    data.forEach(regionData => {
        const canvasContainer = document.createElement('div');
        canvasContainer.classList.add('canvas-container');
        const canvas = document.createElement('canvas');
        canvas.id = `Grafico-${regionData.region}`;
        canvasContainer.appendChild(canvas);
        contenedor.appendChild(canvasContainer);

        let dates = regionData.confirmed.map(entry => entry.date);
        let values = regionData.confirmed.map(entry => parseInt(entry.value));

        let ctx = canvas.getContext('2d');
        new Chart(ctx, {
            type: 'line',
            data: {
                labels: dates,
                datasets: [{
                    label: `Número de Confirmados en ${regionData.region}`,
                    data: values,
                    borderColor: 'rgba(75, 192, 192, 1)',
                    backgroundColor: 'rgba(75, 192, 192, 0.2)',
                    fill: true,
                }]
            }
        });
    });
}

function getRandomColor(alpha = 1) {
    const r = Math.floor(Math.random() * 255);
    const g = Math.floor(Math.random() * 255);
    const b = Math.floor(Math.random() * 255);
    return `rgba(${r}, ${g}, ${b}, ${alpha})`;
}
```

- Página



- Resultado

5.6. Visualice un gráfico comparativo del crecimiento en regiones excepto Lima y Callao

- HTML

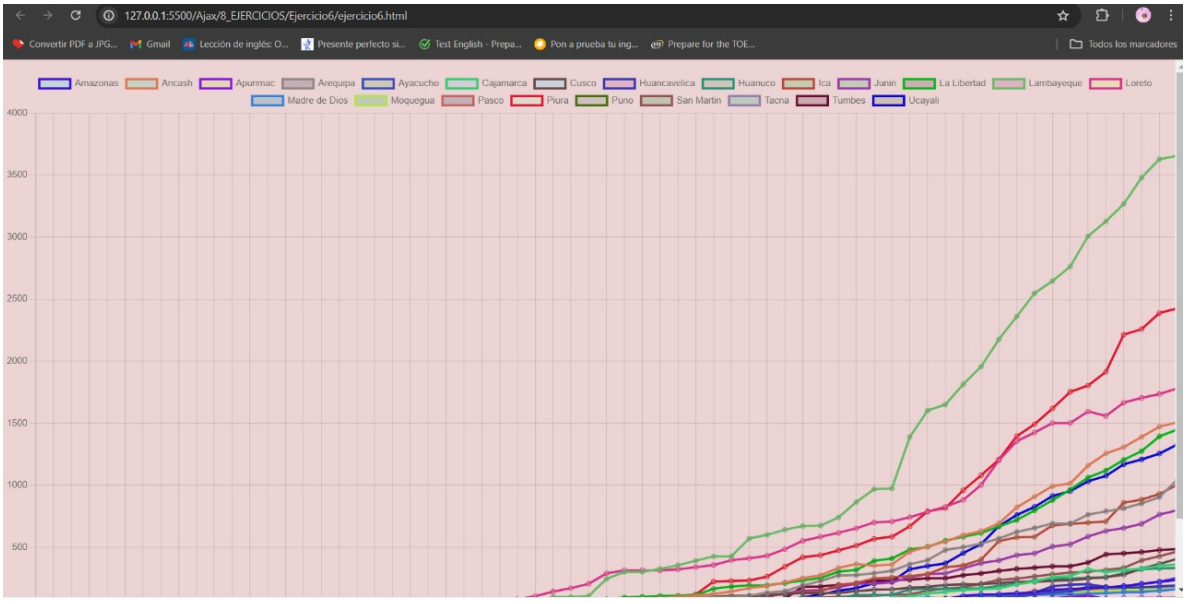
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Comparativa de Regiones</title>
    <link rel="stylesheet" href="style.css">
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <div id="datos">
        <h1>Grafico Comparativo de Regiones (exepto Lima y Callao)</h1>
        <button id="mostrarGrafico">Mostrar Grafico</button>
        <div class="contenedor" style="display: none;">
            <canvas id="GraficoComparativo" width="800" height="400"></canvas>
        </div>
    </div>

    <script src="ejercicio6.js"></script>
</body>
</html>
```

- Script

```
document.addEventListener("DOMContentLoaded", function() {
    const btnMostrarGrafico = document.getElementById('mostrarGrafico');
    const contenedor = document.querySelector('.contenedor');

    btnMostrarGrafico.addEventListener('click', () => {
        let xhr = new XMLHttpRequest();
        xhr.open("GET", "../data.json", true);
        xhr.onload = function() {
            if (xhr.status === 200) {
                let data = JSON.parse(xhr.responseText);
                let regions = data.filter(region => region.region !== "Lima" && region.region !== "Callao");
                if (regions.length > 0) {
                    mostrarGraficoComparativo(regions);
                    contenedor.style.display = 'flex';
                    btnMostrarGrafico.style.display = 'none';
                } else {
                    console.error("No se encontraron datos de regiones");
                }
            } else {
                console.error('Error en la red:', xhr.statusText);
            }
        };
        xhr.onerror = function() {
            console.error('Error en la red');
        };
        xhr.send();
    })
})

function mostrarGraficoComparativo(regions) {
    let labels = [];
    let datasets = [];

    regions.forEach(region => {
        let dates = region.confirmed.map(entry => entry.date);
        let values = region.confirmed.map(entry => parseInt(entry.value));

        if (labels.length === 0) {
            labels = dates;
        }

        datasets.push({
            label: region.region,
            data: values,
            borderColor: getRandomColor(),
            backgroundColor: getRandomColor(0.2),
            fill: false,
        })
    });
    let ctx = document.getElementById('GraficoComparativo').getContext('2d');
    new Chart(ctx, {
        type: 'line',
        data: {
            labels: labels,
            datasets: datasets
        }
    });

}
function getRandomColor(alpha = 1) {
    const r = Math.floor(Math.random() * 255);
    const g = Math.floor(Math.random() * 255);
    const b = Math.floor(Math.random() * 255);
    return `rgba(${r}, ${g}, ${b}, ${alpha})`;
}
```

- Página

- Resultado



5.7. Haga gráficos comparativos entre regiones elegidas por el usuario.

- HTML

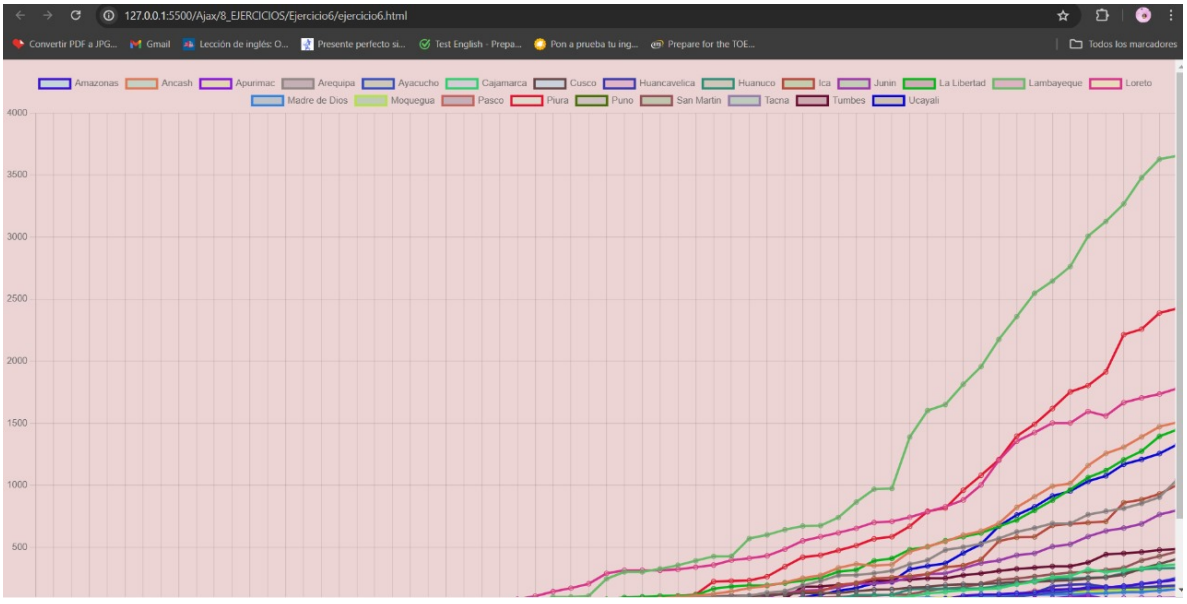
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Comparativa de Regiones</title>
    <link rel="stylesheet" href="style.css">
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <div id="datos">
        <h1>Grafico Comparativo de Regiones (exepto Lima y Callao)</h1>
        <button id="mostrarGrafico">Mostrar Grafico</button>
        <div class="contenedor" style="display: none;">
            <canvas id="GraficoComparativo" width="800" height="400"></canvas>
        </div>
    </div>

    <script src="ejercicio6.js"></script>
</body>
</html>
```

- Script

```
document.addEventListener("DOMContentLoaded", function() {
    const btnMostrarGrafico = document.getElementById('mostrarGrafico');
    const contenedor = document.querySelector('.contenedor');

    btnMostrarGrafico.addEventListener('click', () => {
        let xhr = new XMLHttpRequest();
        xhr.open("GET", "../data.json", true);
        xhr.onload = function() {
            if (xhr.status === 200) {
                let data = JSON.parse(xhr.responseText);
                let regions = data.filter(region => region.region !== "Lima" && region.region !== "Callao");
                if (regions.length > 0) {
                    mostrarGraficoComparativo(regions);
                    contenedor.style.display = 'flex';
                    btnMostrarGrafico.style.display = 'none';
                } else {
                    console.error("No se encontraron datos de regiones");
                }
            } else {
                console.error('Error en la red:', xhr.statusText);
            }
        };
        xhr.onerror = function() {
            console.error('Error en la red');
        };
        xhr.send();
    })
})

function mostrarGraficoComparativo(regions) {
    let labels = [];
    let datasets = [];

    regions.forEach(region => {
        let dates = region.confirmed.map(entry => entry.date);
        let values = region.confirmed.map(entry => parseInt(entry.value));

        if (labels.length === 0) {
            labels = dates;
        }

        datasets.push({
            label: region.region,
            data: values,
            borderColor: getRandomColor(),
            backgroundColor: getRandomColor(0.2),
            fill: false,
        })
    });
    let ctx = document.getElementById('GraficoComparativo').getContext('2d');
    new Chart(ctx, {
        type: 'line',
        data: {
            labels: labels,
            datasets: datasets
        }
    });

}
function getRandomColor(alpha = 1) {
    const r = Math.floor(Math.random() * 255);
    const g = Math.floor(Math.random() * 255);
    const b = Math.floor(Math.random() * 255);
    return `rgba(${r}, ${g}, ${b}, ${alpha})`;
}
```

- Página

- Resultado



5.8. Visualice un gráfico comparativo del crecimiento en regiones excepto Lima y Callao, mostrando el número de confirmados por cada día

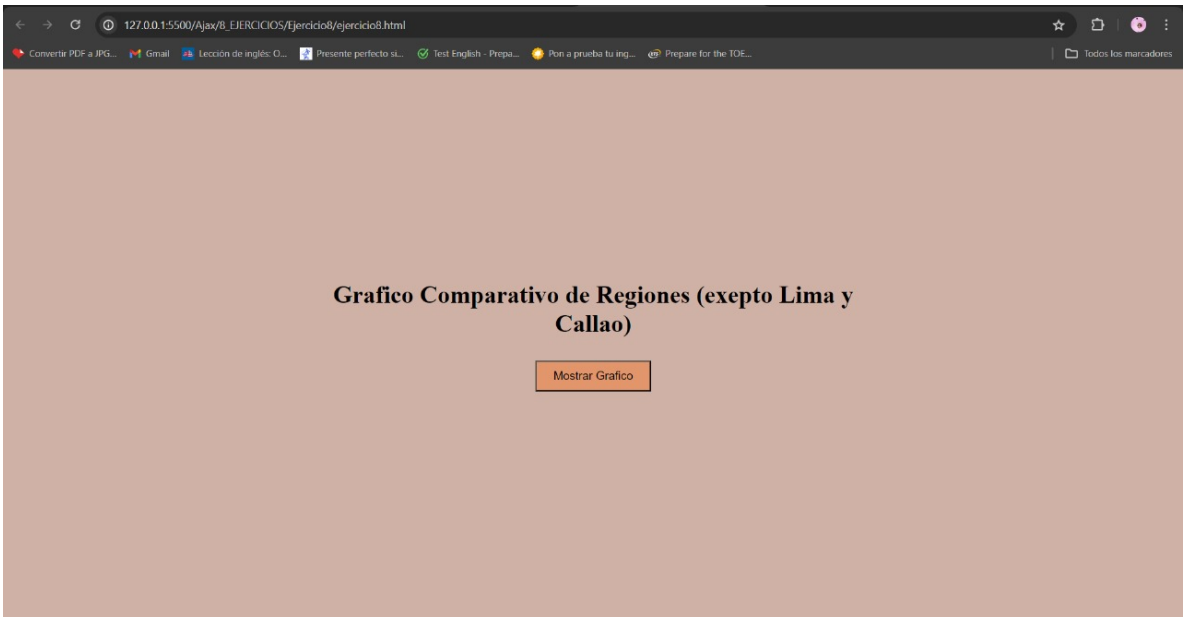
- HTML

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Comparativa de Regiones</title>
    <link rel="stylesheet" href="style.css">
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <div id="datos">
        <h1>Grafico Comparativo de Regiones (exepto Lima y Callao)</h1>
        <button id="mostrarGrafico">Mostrar Grafico</button>
        <div class="contenedor" style="display: none;">
            <canvas id="GraficoComparativo" width="800" height="400"></canvas>
        </div>
    </div>

    <script src="ejercicio8.js"></script>
</body>
</html>
```

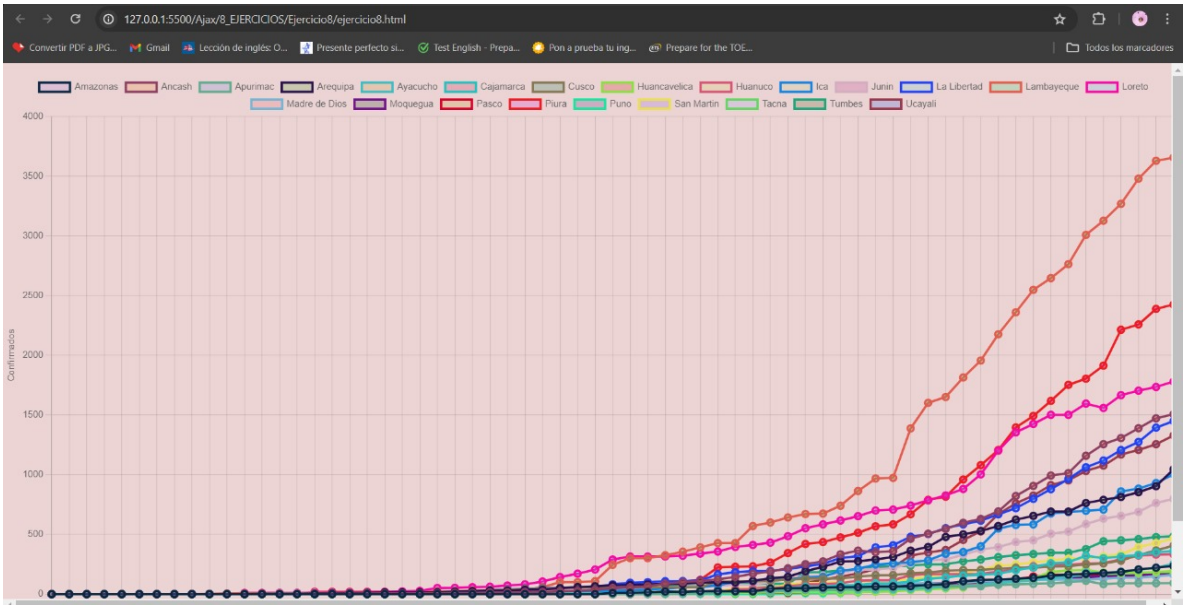
- Script

```
document.addEventListener("DOMContentLoaded", function() {
    const btnMostrarGrafico = document.getElementById('mostrarGrafico');
    const contenedor = document.querySelector('.contenedor');

    btnMostrarGrafico.addEventListener('click', () => {
        let xhr = new XMLHttpRequest();
        xhr.open("GET", "../data.json", true);
        xhr.onload = function() {
            if (xhr.status === 200) {
                let data = JSON.parse(xhr.responseText);
                let regions = data.filter(region => region.region !== "Lima" && region.region !== "Callao");
                if (regions.length > 0) {
                    mostrarGraficoComparativo(regions);
                    contenedor.style.display = "flex";
                    btnMostrarGrafico.style.display = 'none';
                } else {
                    console.error("No se encontraron datos de regiones");
                }
            } else {
                console.error('Error en la red:', xhr.statusText);
            }
        };
        xhr.onerror = function() {
            console.error('Error en la red');
        };
        xhr.send();
    })
})

function mostrarGraficoComparativo(regions) {
    let labels = [];
    let datasets = [];

    regions.forEach((region, index) => {
        let dates = region.confirmed.map(entry => entry.date);
        let values = region.confirmed.map(entry => parseInt(entry.value));

        if (labels.length === 0) {
            labels = dates;
        }

        datasets.push({
            label: region.region,
            data: values,
            borderColor: getRandomColor(),
            backgroundColor: getRandomColor(0.2),
            fill: false,
            pointRadius: 4,
            borderWidth: 3,
        })
    });

    let ctx = document.getElementById('GraficoComparativo').getContext('2d');
    new Chart(ctx, {
        type: 'line',
        data: {
            labels: labels,
            datasets: datasets
        },
        options: {
            scales: {
                x: {
                    display: true,
                    title: {
                        display: true,
                        text: 'Fecha'
                    }
                },
                y: {
                    display: true,
                    title: {
                        display: true,
                        text: 'Confirmados'
                    },
                    suggestedMin: 0,
                }
            },
            title: {
                display: true,
                text: 'Crecimiento de Confirmados en Regiones (excepto Lima y Callao)',
                fontSize: 18,
                fontColor: '#333'
            },
            legend: {
                display: true,
                position: 'bottom',
                labels: {
                    fontColor: '#333'
                }
            },
            plugins: {
                legend: {
                    display: true,
                    position: 'top'
                },
                tooltip: {
                    enabled: true,
                    mode: 'index', //Se muestra todos los casos del día a la vez
```

- Página



- Resultado



6. Rúbricas

6.1. Entregable Informe

Tabla 1: Tipo de Informe

Informe	
Latex	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y facil de leer.

6.2. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumplio con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobada, siempre y cuando cumpla con todos lo items.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Tabla 2: Niveles de desempeño

	Nivel			
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Tabla 3: Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	2	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	1	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	2	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
6. Fechas	Las fechas de modificación del código fuente estan dentro de los plazos de fecha de entrega establecidos.	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	1	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	X	1	
Total		20		13	

7. Referencias

- https://www.w3schools.com/java/default.asp
- https://www.geeksforgeeks.org/insertion-sort/