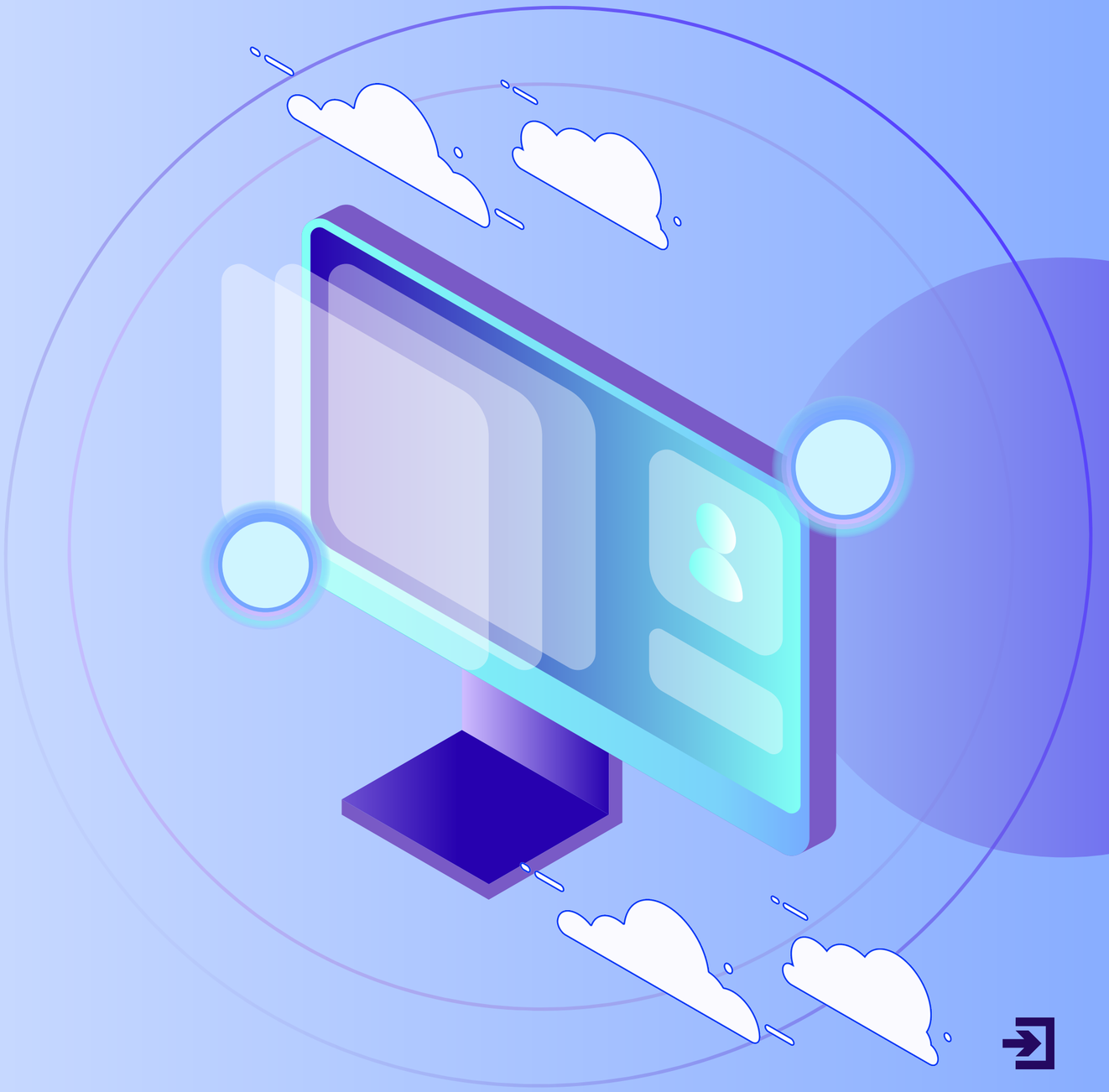
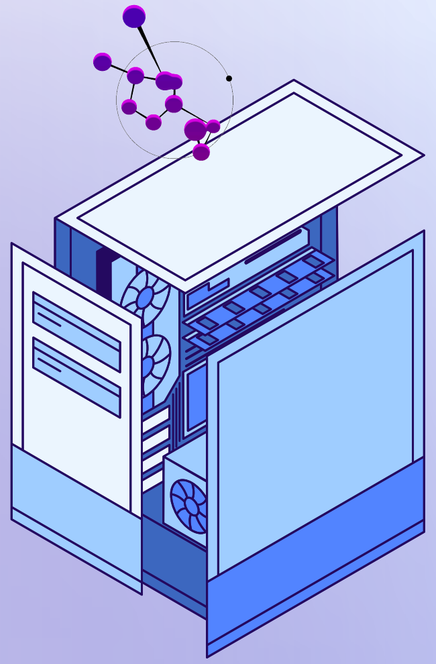
Sistemas operativos

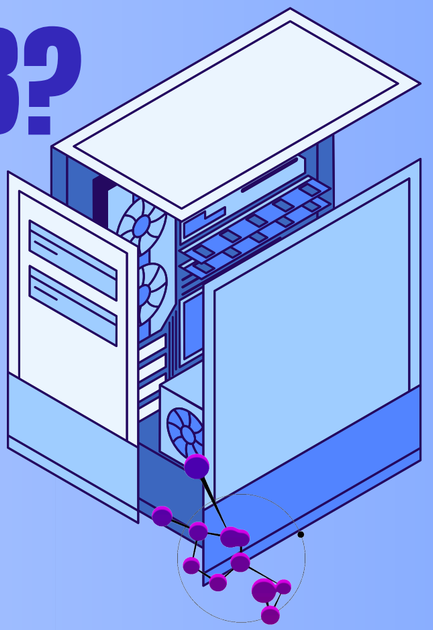
API PARA ANALIZAR EL RENDIMIENTO DE APLICACIONES WEB

Sebastian Ortiz, Jaime Muñoz y Estefania G



¿POR QUÉ ES IMPORTANTE ANALIZAR EL RENDIMIENTO DE APLICACIONES WEB?





Afecta la experiencia del usuario y su retención.
Las aplicaciones lentas generan frustración y pérdida de usuarios.

Objetivo:
Comparar métricas clave como tiempo de respuesta, CPU y memoria.

METODOLOGIA DE DESAROLLO

FRONTEND

- React.js y Tailwind CSS para la interfaz.
- Gráficos interactivos generados con Chart.js.

API(BACKEND)

- Implementado en Node.js.
- Uso de middleware como morgan y cors.
- Recolección de métricas clave con la biblioteca axios.

CONFIGURACIÓN DE LA API

```
const app = express();

// Habilitar CORS
app.use(cors());

// Middleware para registrar las solicitudes
app.use(morgan('dev'));

// Función para medir rendimiento de una URL
const measurePerformance = async (url) => {
    const startTime = Date.now();

    try {
        const response = await axios.get(url);
        const endTime = Date.now();
        const executionTime = endTime - startTime;

        // Obtener uso de CPU y memoria
        const cpuUsage = process.cpuUsage();
        const memoryUsage = process.memoryUsage();

        return {
            url,
            responseTime: `${executionTime}ms`,
            statusCode: response.status,
            cpuUsage: cpuUsage,
            memoryUsage: {
                rss: `${(memoryUsage.rss / 1024 / 1024).toFixed(2)} MB`,
                heapTotal: `${(memoryUsage.heapTotal / 1024 / 1024).toFixed(2)} MB`,
                heapUsed: `${(memoryUsage.heapUsed / 1024 / 1024).toFixed(2)} MB`,
            }
        };
    } catch (error) {
        return {
            url,
            error: 'Error measuring performance',
            message: error.message
        };
    }
};
```

ENDPOINT PARA MEDIR EL RENDIMIENTO DE LAS APLICACIONES WEB

```
app.get('/performance', async (req, res) => {
    const url1 = req.query.url1;
    const url2 = req.query.url2;

    if (!url1 || !url2) {
        return res.status(400).json({ error: 'Please provide two valid URLs to test performance' });
    }

    try {
        // Ejecutar ambas solicitudes de manera asíncrona
        const [result1, result2] = await Promise.all([measurePerformance(url1),
measurePerformance(url2)]);

        res.json({
            message: 'Performance analyzed successfully',
            results: {
                url1: result1,
                url2: result2
            }
        });
    } catch (error) {
        res.status(500).json({
            message: 'Error measuring performance',
            error: error.message
        });
    }
});
```

DEMO

CONCLUSION

- **Herramienta eficaz**: Análisis, mejora.
- **Impacto positivo:** Optimización, experiencia.
- **Resultados claros:** Diferencias, decisiones.
- **Futuro prometedor**: Monitoreo, escalabilidad.