Config\logger.js

```
// Importamos el módulo winston, que nos permite crear un logger
const winston = require('winston');

// Creamos un formato personalizado para los logs
const customFormat = winston.format.printf(({ level, message, timestamp }) => {
    // El formato del log es 'timestamp - LEVEL: message'
    return `${timestamp} - ${level.toUpperCase()}: ${message}`;
});

// Creamos un nuevo logger con la función createLogger de winston
const logger = winston.createLogger({
    // Especificamos el nivel de log. Los niveles disponibles son: error, warn, info, verbose, debug y silly
    level: 'info',
    // Especificamos el formato de los logs. Estamos utilizando una combinación de timestamp y el formato personalizado
    format: winston.format.combine(
        // Agregamos un timestamp a cada log. El formato del timestamp es 'YYYY-MM-DD - HH:mm'
        winston.format.timestamp({
            format: 'YYYY-MM-DD - HH:mm:ss'
        }),
        // Utilizamos el formato personalizado para los logs
        customFormat
    ),
    // Especificamos los transportes, que determinan dónde se guardan los logs
    transports: [
        // Guardamos los logs de nivel error en el archivo error.log
        new winston.transports.File({ filename: 'error.log', level: 'error' }),
        // Guardamos todos los logs en el archivo combined.log
        new winston.transports.File({ filename: 'combined.log' }),
    ],
});

// Si no estamos en el entorno de producción, también mostramos los logs en la consola
if (process.env.NODE_ENV !== 'production') {
    logger.add(new winston.transports.Console({
        // Para los logs en la consola, también utilizamos el formato personalizado
        format: winston.format.combine(
            winston.format.colorize(),
            customFormat
        ),
    }));
}

// Exportamos el logger para que pueda ser utilizado en otras partes de la aplicación
module.exports = logger;
```