Guía de Archivos Base en un Proyecto Cells

Esta guía resume los archivos que normalmente se incluyen en un **scaffold** generado por DevOps al crear un nuevo repositorio. Cada pieza cumple una función específica para asegurar calidad, mantenibilidad y consistencia.

Archivos de configuración

.gitignore

- Define qué archivos o carpetas **no deben versionarse** en Git.
- Ejemplo:

```
node_modules/
  dist/
  .env
  *.log
```

- **Beneficio:** evita subir archivos innecesarios o sensibles al repositorio.

ESLint (.eslintrc.*)

- Herramienta para analizar y mantener un estilo consistente en **JavaScript/TypeScript**.
- Detecta errores comunes y malas prácticas.
- Ejemplo:

```
{
    "extends": "eslint:recommended",
    "env": { "es2021": true, "node": true },
    "rules": {
       "semi": ["error", "always"],
       "quotes": ["error", "double"]
    }
  }
```

- **Beneficio:** asegura legibilidad y reduce errores en el código.

Stylelint (stylelintrc.json)

- Similar a ESLint, pero para **CSS/SCSS**.
- Ayuda a mantener estilos consistentes y evitar malas prácticas en hojas de estilo.
- Ejemplo:

```
module.exports = {
    extends: "stylelint-config-standard",
    rules: {
        "indentation": 2,
        "string-quotes": "double"
    }
};
```

- **Beneficio:** facilita mantener un diseño limpio y estandarizado.

sonar-project.properties

- Configuración para **SonarQube**, herramienta de análisis estático.
- Ejemplo:

```
sonar.projectKey=mi-proyecto
sonar.projectName=Mi Proyecto
sonar.sources=src
sonar.tests=tests
sonar.sourceEncoding=UTF-8
```

- **Beneficio:** detecta errores, vulnerabilidades, code smells y mide la cobertura de pruebas.

Documentación

README.md

- Documentación mínima de cada proyecto.
- Incluye:
    - Descripción breve.
    - Requisitos previos.
    - Pasos de instalación.
    - Comandos de ejecución.
- Ejemplo:

````
# Mi Proyecto

Aplicación para gestionar tareas.

## Instalación
```bash
npm install

## Ejecución

npm start

```
````

CHANGELOG.md

- Registro de cambios del proyecto, por versión.
- Ejemplo (formato Keep a Changelog):

````
## [1.1.0] - 2025-09-30
### Added
- Nueva función de exportar reportes.
### Fixed
- Error en validación de fechas.
````

- **Beneficio:** transparencia en la evolución del software.
- Guía oficial para mantener un changelog
- Guía oficial para archivos Markdown

Automatización y CI/CD

package.json

- Archivo principal en proyectos **Node.js**.
- Contiene:
    - **Dependencias**.
    - **Scripts de npm** (build, test, lint, etc.).
    - **Versión** del proyecto (semántica: major.minor.patch).
- Ejemplo:

```
{
  "name": "mi-proyecto",
  "version": "1.0.0",
  "scripts": {
    "lint": "eslint 'src/**/*.{js,jsx}'",
    "stylelint": "stylelint 'src/**/*.{css,scss}'",
    "test": "jest"
  }
}
```

Jenkinsfile

- Define el pipeline de **Integración y Despliegue Continuo (CI/CD)**.
- Ejemplo básico:

```
pipeline {
    agent any
    stages {
        stage('Install') {
        steps { sh 'npm install' }
        }
        stage('Lint') {
        steps {
        sh 'npm run lint'
        sh 'npm run stylelint'
        }
        }
    stage('Test') {
        steps { sh 'npm test' }
        }
        stage('SonarQube analysis') {
                steps {
                        withSonarQubeEnv('SonarQube') {
                        sh 'sonar-scanner'
                        }
                }
        }
```

```
        stage('Deploy') {
        steps { sh './deploy.sh' }
        }
    }
}
```

- **Beneficio:** automatiza el ciclo de vida del proyecto (build → test → análisis → deploy).
- Guía oficial Jenkins

Beneficios del scaffold

- Ahorra tiempo al iniciar un proyecto.
- Estandariza buenas prácticas en todos los repos.
- Evita errores comunes.
- Refuerza la calidad, seguridad y mantenibilidad del software.
- Permite que los equipos se enfoquen en **desarrollar funcionalidad**, no en configurar.

Con este set de archivos, cualquier proyecto comienza con una base sólida de **documentación, calidad de código, análisis estático y CI/CD**.