**Jenkins**

1. Adaptar pom.xml con plugins Maven
2. Crear Jenkinsfile
3. Subir a GitHub
4. En Jenkins crear pipeline desde Blue Ocean
5. Si hay errores revisar versión java y los plugins Maven en el pom.xml

**Jenkins-Sonar**

1. Crear proyecto en Sonarcloud.io vinculado al repositorio GitHub
2. Añadir al pom.xml las propiedades sonar, incluyendo la propiedad sonar.login
3. Agregar a Jenkinsfile el comando ‘mvn verify sonar:sonar’
4. Subir a GitHub
5. Ejecutar pipeline en Jenkins

**GitHub Actions**:

1. Vamos a GitHub y en la pestaña Actions seleccionar workflow Java with Maven
2. Creará automáticamente el archivo yml con la configuración del workflow: cambiar versión a java a 16 y hacer commit
3. Ya está todo configurado, cada vez que se hace commit se construye

**GitHub Actions – Sonar**

1. Crear nuevo proyecto sonarcloud.io vinculado al repo
2. En método de análisis seleccionar GitHub Actions
3. Crear secret en GitHub: SONAR\_TOKEN
4. Crear nuevo workflow en el repositorio con el código proporcionado por sonarcloud

Sintaxis Github workflows:

<https://github.com/actions/starter-workflows>

<https://docs.github.com/en/actions>

**GitLab**

<https://docs.gitlab.com/ee/ci/quick_start/index.html>

**Vercel**

<https://vercel.com/dashboard>

Hacer fork de <https://angular-realworld-example-app-rosy.vercel.app/>

Vincular con GitHub y seleccionar el repositorio.

**Netlify**

<https://app.netlify.com>

Comparativa: <https://www.stackshare.io/stackups/netlify-vs-vercel>

**Heroku**

<https://devcenter.heroku.com/categories/java-support>

<https://devcenter.heroku.com/articles/deploying-spring-boot-apps-to-heroku>

Actividad 1: API REST Spring Boot

Actividad 2: API REST Spring Boot + MySQL

Actividad 3: API REST Spring Boot + Login JWT + MySQL

**Herramientas IDEs online:**

<https://github.com/cdr/code-server>

<https://stackblitz.com/>

<https://codesandbox.io/>