



Laboratorio 6

Integración continua con CircleCI y Heroku

Integrantes:

Luisa Fernanda Bermúdez Girón

Karol Daniela Ladino Ladino

Profesor:

Iván Darío Viasus Quintero

Curso:

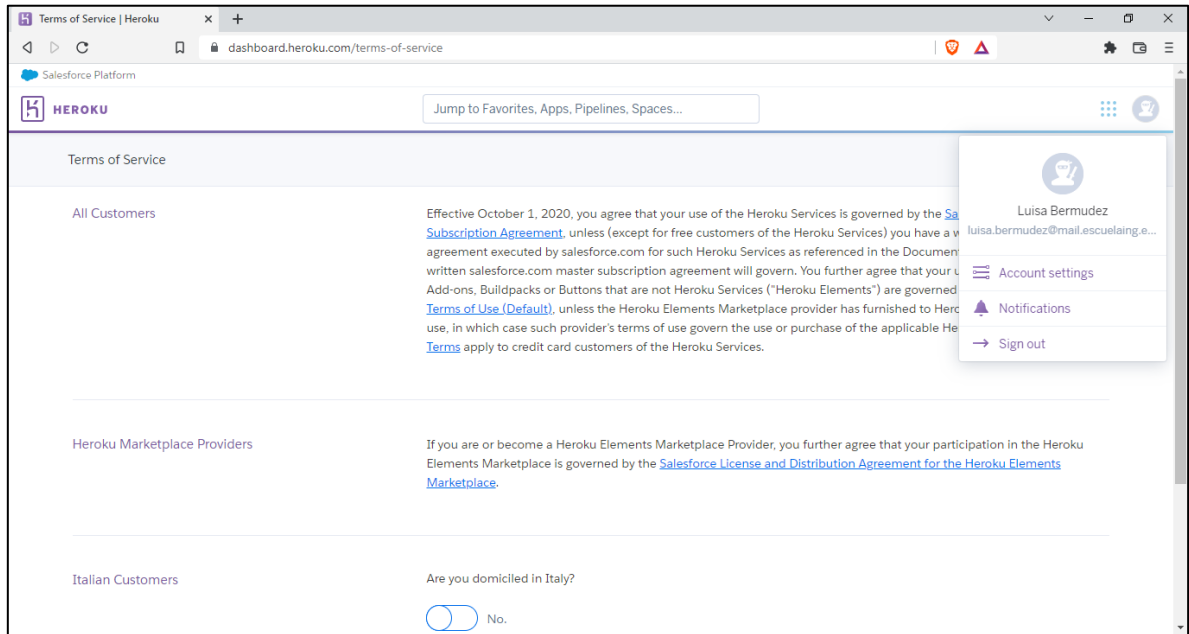
CVDS - 2

Fecha De Entrega:

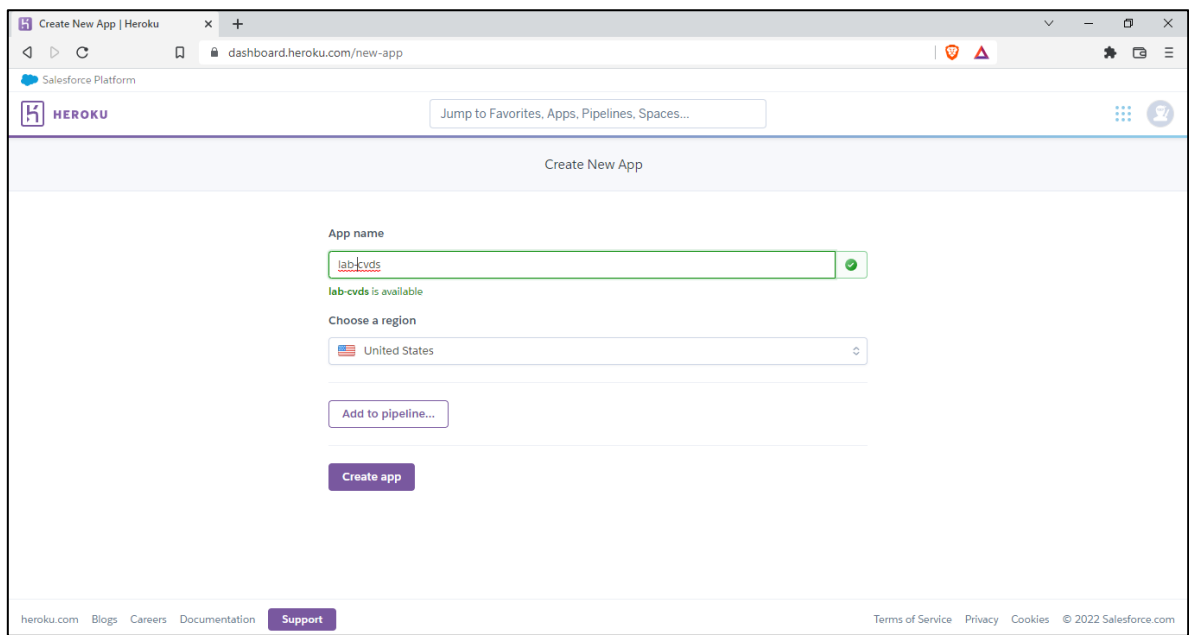
28-09-2022

PARTE I. INTEGRACIÓN CONTINUA

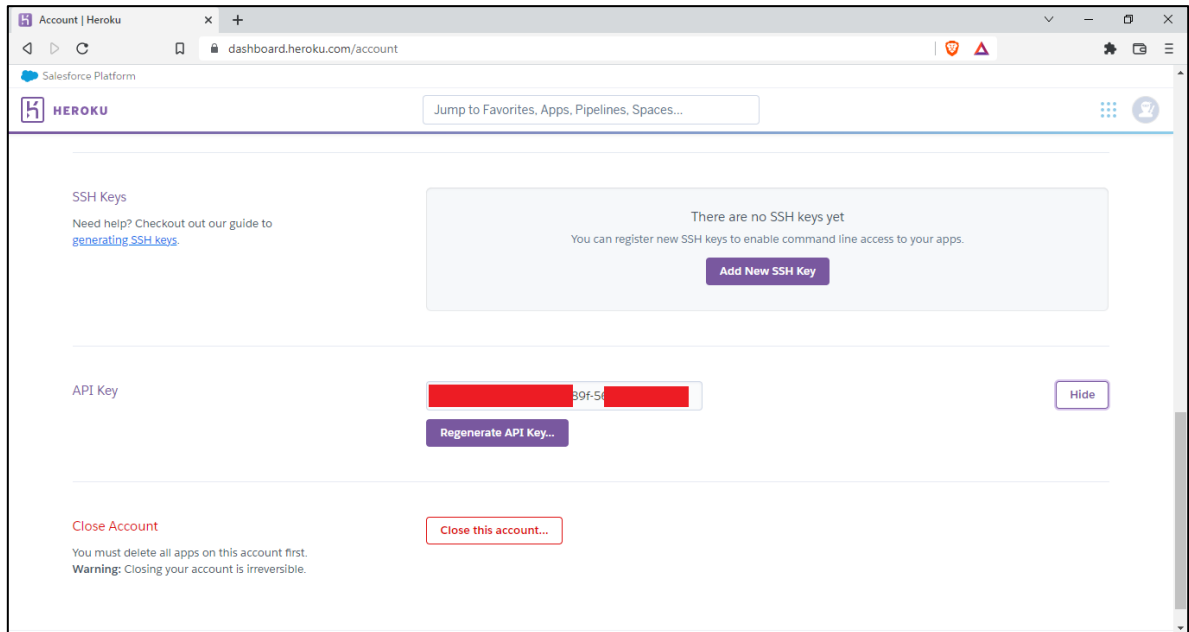
1. Cree (si no la tiene aún) una cuenta en el proveedor PAAS Heroku (www.heroku.com).



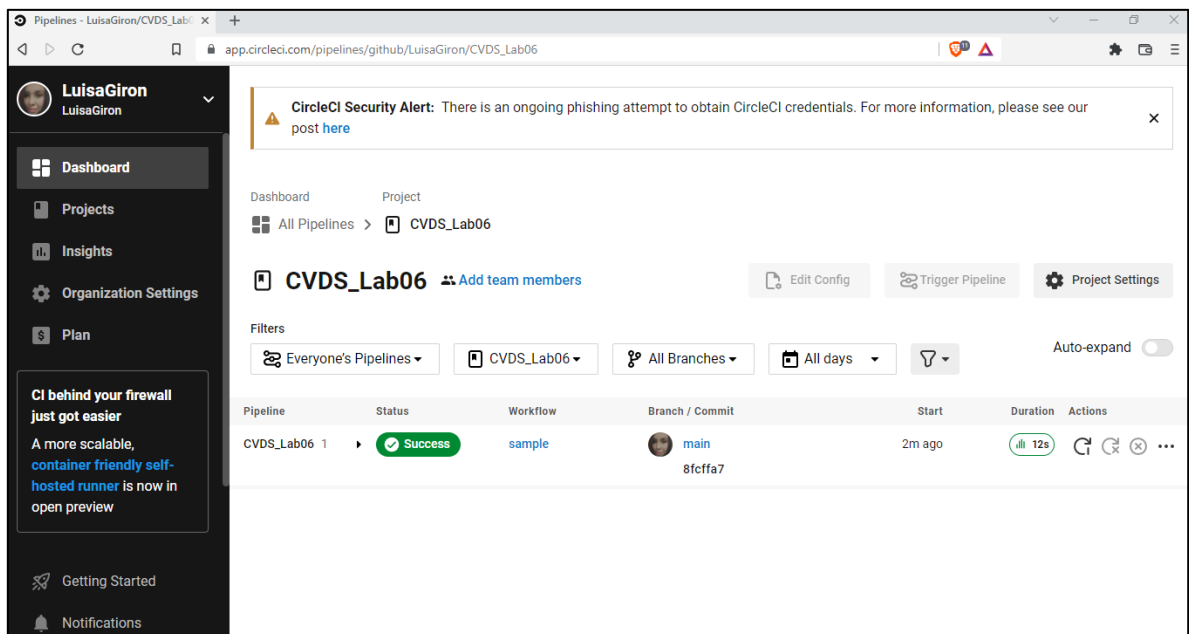
2. Acceda a su cuenta en Heroku y cree una nueva aplicación:



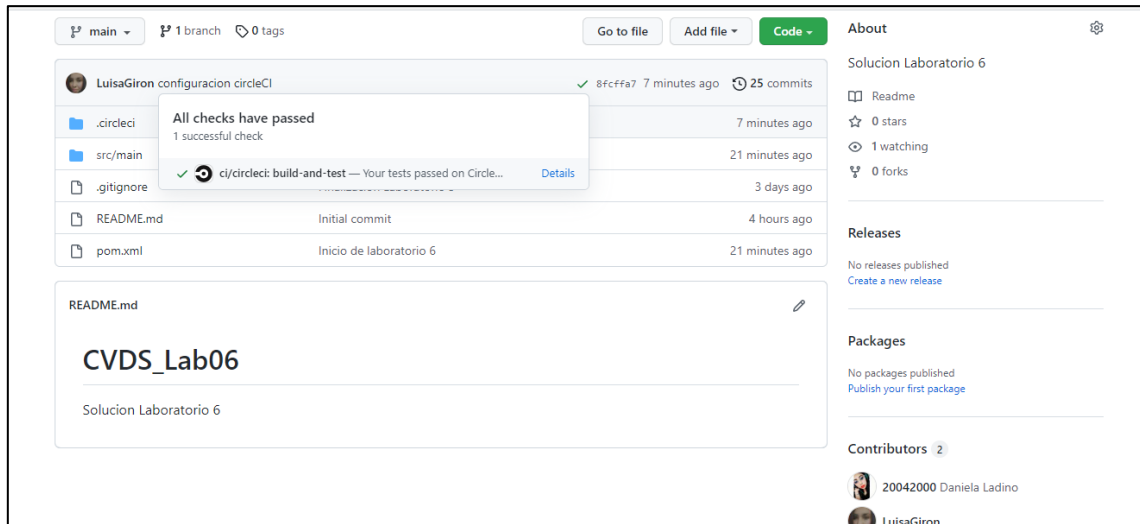
3. Después de crear su cuenta en Heroku y la respectiva aplicación, genere una llave de API:
Opción Manage Account:



4. Ingrese a la plataforma de integración continua Circle.CI (www.circleci.com). Para ingresar, basta que se autentique con su usuario de GitHub.

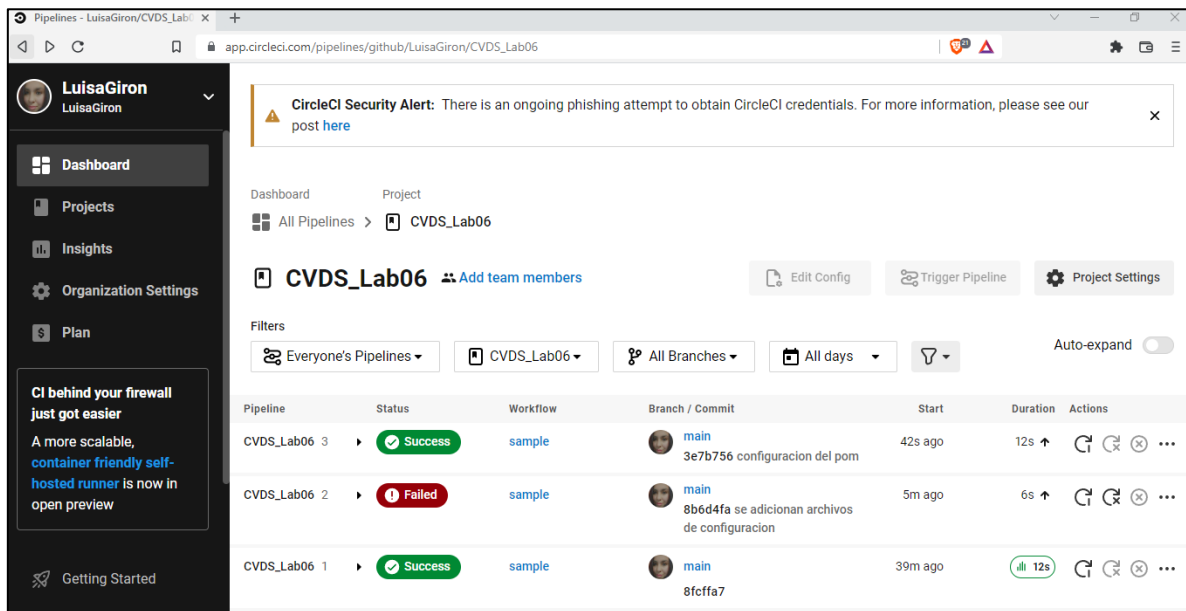


5. Seleccione la opción de agregar proyectos. En la organización o usuario de GitHub, seleccione el proyecto al que le va a hacer despliegue continuo, y haga clic en "Build Project":

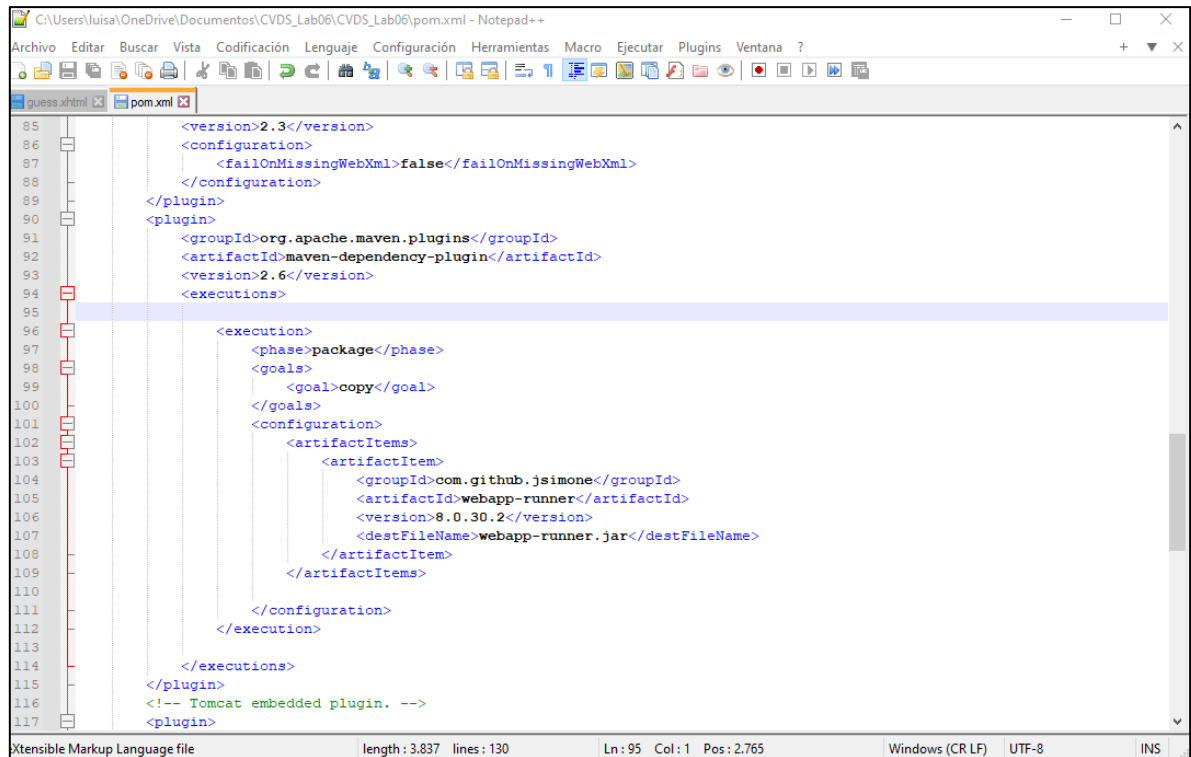


6. Si todo queda correctamente configurado, cada vez que hagan un PUSH al repositorio, CircleCI ejecutará la fase de construcción del proyecto. Para que cuando las pruebas pasen automáticamente se despliegue en Heroku, debe definir en el archivo circle.yml (ubicado en la raíz del proyecto):

- La rama del repositorio de GitHub que se desplegará en Heroku. o El nombre de la aplicación de Heroku en la que se hará el despliegue.
- La ejecución de la fase 'site' de Maven, para generar la documentación de pruebas, cubrimiento de pruebas y análisis estático (cuando las mismas sean habilitadas).



7. Rectifique que en el pom.xml, en la fase de construcción, se tenga el siguiente plugin (es decir, dentro de <build><plugins>):



```
85 <version>2.3</version>
86 <configuration>
87 <failOnMissingWebXml>false</failOnMissingWebXml>
88 </configuration>
89 </plugin>
90 <plugin>
91 <groupId>org.apache.maven.plugins</groupId>
92 <artifactId>maven-dependency-plugin</artifactId>
93 <version>2.6</version>
94 <executions>
95
96 <execution>
97 <phase>package</phase>
98 <goals>
99 <goal>copy</goal>
100 </goals>
101 <configuration>
102 <artifactItems>
103 <artifactItem>
104 <groupId>com.github.jsimone</groupId>
105 <artifactId>webapp-runner</artifactId>
106 <version>8.0.30.2</version>
107 <destFileName>webapp-runner.jar</destFileName>
108 </artifactItem>
109 </artifactItems>
110 </configuration>
111 </execution>
112 </executions>
113 </plugin>
114 <!-- Tomcat embedded plugin. -->
115 <plugin>
```

8. Heroku requiere los siguientes archivos de configuración (con sus respectivos contenidos) en el directorio raíz del proyecto, de manera que sea qué versión de Java utilizar, y cómo iniciar la aplicación, respectivamente:

system.properties


```
java.runtime.version=1.8
```

Procfile

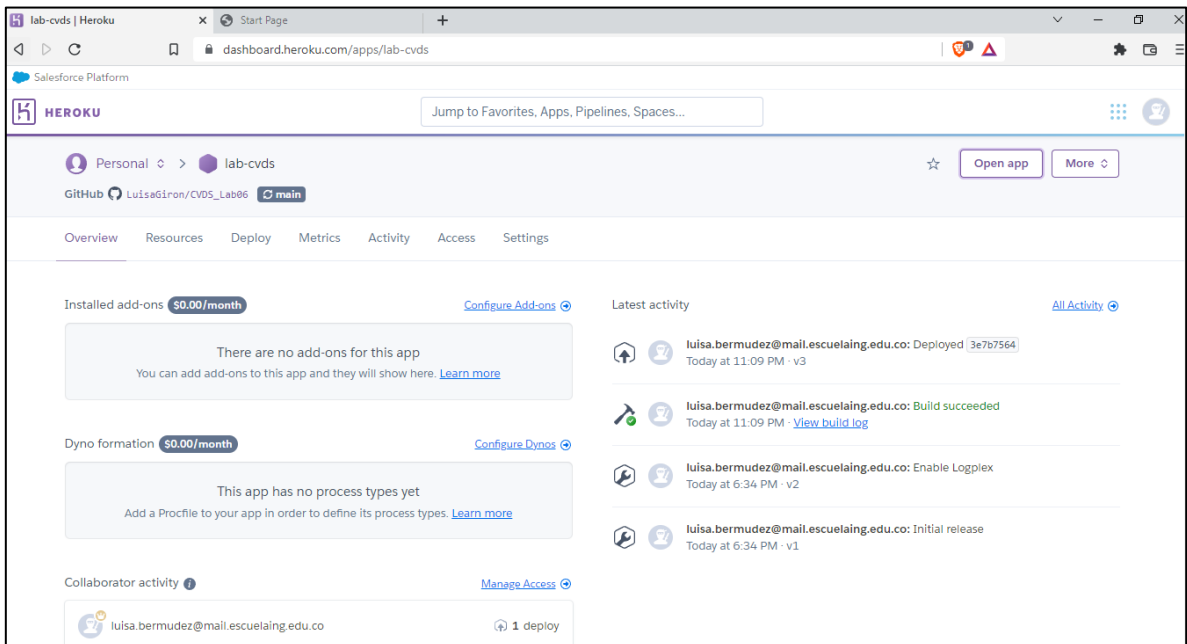
```
web: java $JAVA_OPTS -jar target/dependency/webapp-runner.jar --port $PORT target/*.war
```

.circleci	01/10/2022 04:56 p. m.	Carpeta de archivos	
.git	01/10/2022 04:56 p. m.	Carpeta de archivos	
src	01/10/2022 04:56 p. m.	Carpeta de archivos	
.gitignore	01/10/2022 04:56 p. m.	Documento de te...	1 KB
pom.xml	01/10/2022 04:56 p. m.	Documento XML	4 KB
Procfile	01/10/2022 04:56 p. m.	Archivo	1 KB
README.md	01/10/2022 04:56 p. m.	Archivo de origen ...	1 KB
system.properties	01/10/2022 04:56 p. m.	Archivo de origen ...	1 KB

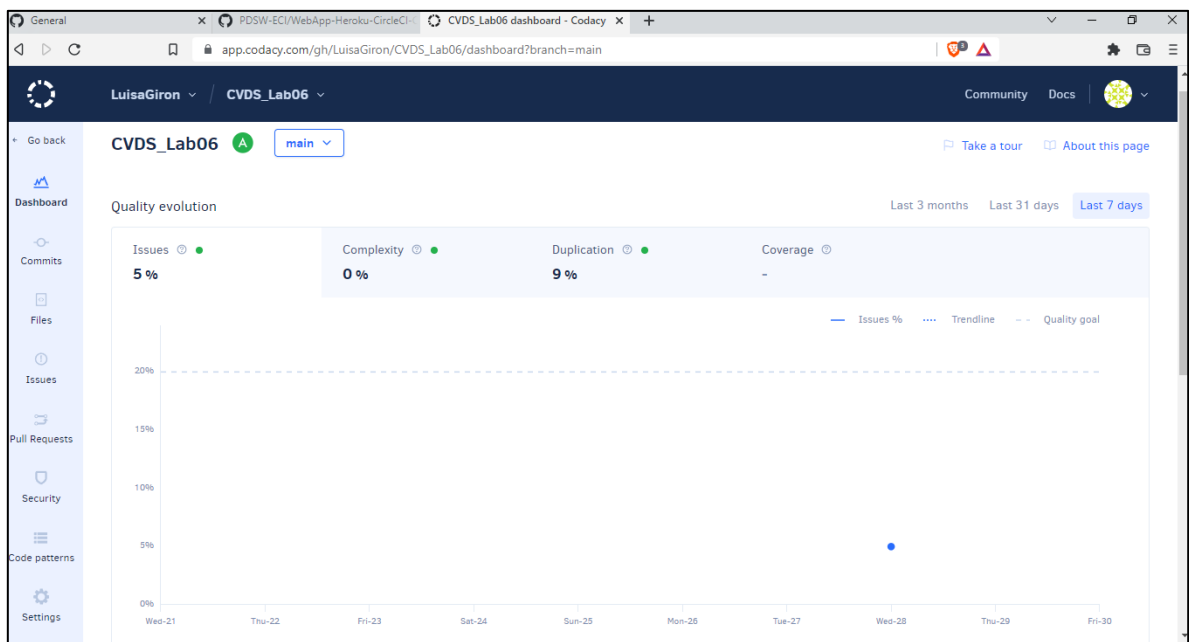
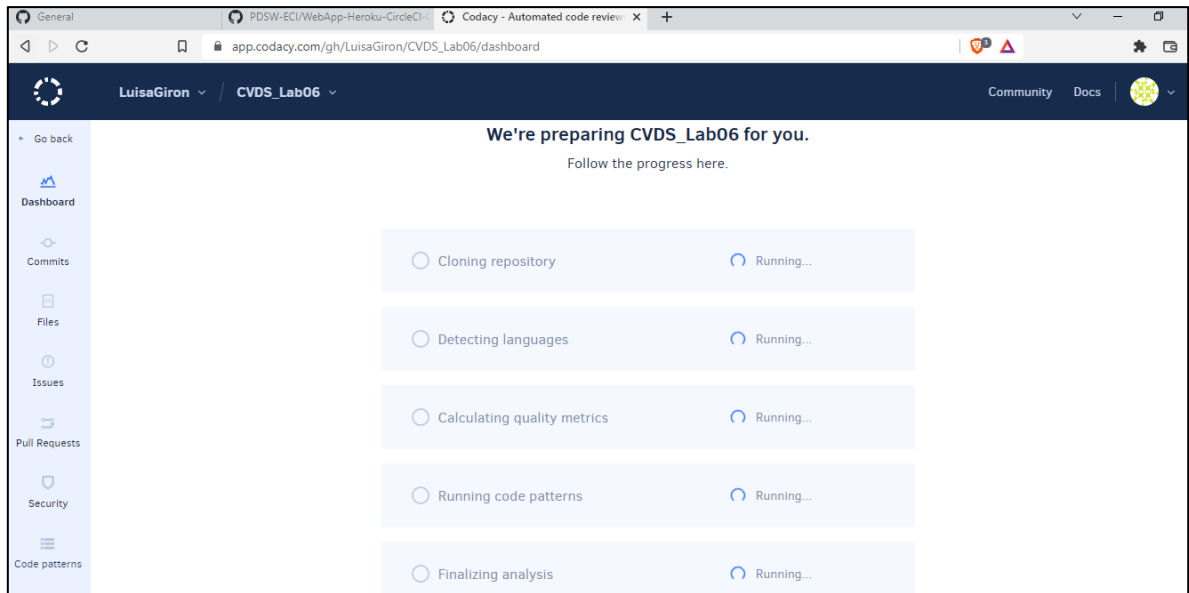
9. El ambiente de despliegue continuo requiere también un archivo de configuración 'circle.yml' en la raíz del proyecto, en el cual se indica (entre otras cosas) en qué aplicación de Heroku se debe desplegar la aplicación que está en GitHub. Puede basarse en el siguiente archivo, teniendo en cuenta que se debe ajustar el parámetro 'appname': <https://github.com/PDSW-ECI/base-proyectos/blob/master/circle.yml>

.circleci			
Nombre	Fecha de modificación	Tipo	Tamaño
 config.yml	01/10/2022 04:56 p. m.	Archivo de origen ...	2 KB

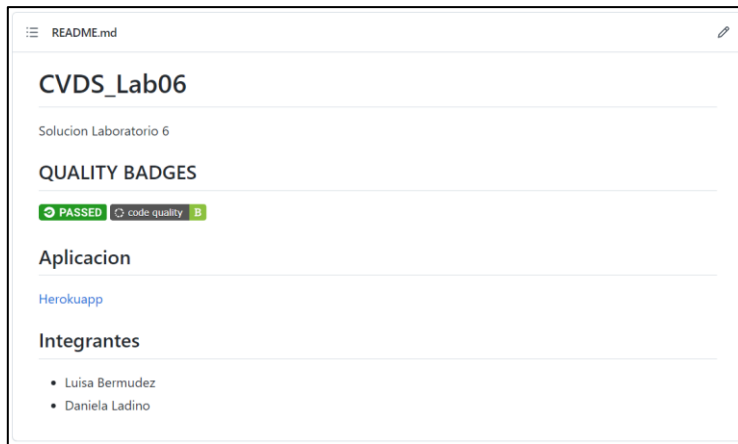
10. Haga commit y push de su repositorio local a GitHub. Abra la consola de CircleCI y verifique que el de descarga, compilación, y despliegue. Igualmente, verifique que la aplicación haya sido desplegada en Heroku.



11. Ahora, va a integrar un entorno de Análisis de Calidad de Código a su proyecto, el cual detecte continuamente defectos asociados al mismo. Auténtíquese en [CODACY](https://app.codacy.com) con su cuenta de GitHub, y agregue el proyecto antes creado.



12. Cree un archivo README.md para su proyecto, y asocie al mismo dos 'badges', que permitan conocer el estado del proyecto en cualquier momento: uno para [Circle.CI](#), y otro para [CODACY](#). El proyecto usado como referencia, ya incluye dichos 'badges' en su archivo README: <https://github.com/PDSW-ECI/base-proyectos>



PARTE II. INTERFAZ GRÁFICA

Agregue mayor detalle a la interfaz gráfica de la página web para que se vea más amigable al usuario. Busque y agregue estilos, colores de fondo, títulos, centrar la página, etc. cualquier elemento o detalle que pueda mejorar cómo se ve la página y como el usuario la percibe.

Si no recuerda cómo hacerlo, puede revisar la información disponible en W3Schools para HTML y CSS.



url repositorio: https://github.com/LuisaGiron/CVDS_Lab06.git