



**UNIVERSIDAD
CATÓLICA
DE CÓRDOBA**
JESUITAS

**Ingeniería de Software 3
TRABAJO PRÁCTICO N°12**

Profesor: Ing. Fernando Bono

Alumno: Menel Angelo (1804789)

- 1) Instalo la línea de comando heroku, mostrado en la consigna del TP12.
- 2) Nos registramos con la línea de comando heroku login:

```
~/Ing_3 main heroku login
> Warning: heroku update available from 7.62.0 to 7.65.0.
heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/cli/browser/ef2cd919-e1c5-4937-9fc8-4b42527470c9?requestor=SFMyNTY.g2gDbQAAAA4x0TAuMjIjEwMG4GANCcQBgEAWIAAVGA.eHyp3zJMj0h58gR4PKf9WwWuzRfDj5B-mDme4i_A0Y
Logging in... done
Logged in as angelomenel@hotmail.com
```

```
~/spring-boot main !7 heroku container:login
> Warning: heroku update available from 7.62.0 to 7.65.0.
WARNING! Your password will be stored unencrypted in /home/angelo/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

- 3) Se modifica el dockerfile del repo spring-boot;

```
FROM openjdk:8-jre-alpine

EXPOSE 8080

RUN mkdir /app
COPY --from=MAVEN_TOOL_CHAIN /tmp/target/*.jar /app/spring-boot-application.jar

ENV PORT=8080

ENV JAVA_OPTS="-Xms32m -Xmx128m"

CMD ["java", "-Xms32m", "-Xmx128m", "-jar", "-Dserver.port=${PORT}", "-Djava.security.egd=file:/dev/./urandom", "/app/spring-boot-application.jar"]

ENTRYPOINT exec java $JAVA_OPTS -Djava.security.egd=file:/dev/./urandom -jar /app/spring-boot-application.jar

HEALTHCHECK --interval=1m --timeout=3s CMD wget -q -T 3 -s http://localhost:8080/actuator/health/ || exit 1
```

- 4) Entramos en el repositorio y creamos una aplicación heroku:

```
~/spring-boot main !7 heroku create
> Warning: heroku update available from 7.62.0 to 7.65.0.
Creating app... done, ● desolate-wave-51529
https://desolate-wave-51529.herokuapp.com/ | https://git.heroku.com/desolate-wave-51529.git
```

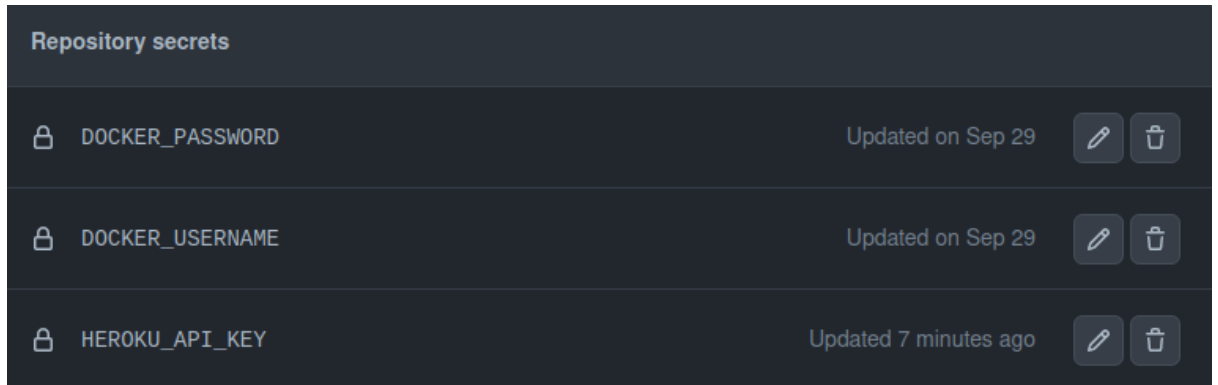
Se crea una aplicación con un nombre random, en este caso:
desolate-wave-51529.

- 5) Generamos y subimos la imagen de Docker al registry de Heroku:

```
~/spring-boot main !7 heroku container:release web --app=desolate-wave-51529
> Warning: heroku update available from 7.62.0 to 7.65.0.
Releasing images web to desolate-wave-51529... done
```

- 6) Si entramos en <https://desolate-wave-51529.herokuapp.com>, podemos ver que la aplicación esta levantada:

- 7) En vez de usar jenkins, utilizo Github Actions para la automatizacion de heroku, para ello primero configuro las credenciales de heroku en Github:



- 8) Luego procedemos a crear el script para heroku:

```

name: Heroku

# Controls when the workflow will run
on:
  # Triggers the workflow on push or pull request events but only for the main branch
  push:
    paths:
      - '/spring-boot/**'
    branches: [ main ]
  pull_request:
    paths:
      - '/spring-boot/**'
    branches: [ main ]

# Allows you to run this workflow manually from the Actions tab
workflow_dispatch:

# A workflow run is made up of one or more jobs that can run sequentially or in parallel
jobs:
  # This workflow contains a single job called "build"
  build:
    # The type of runner that the job will run on
    runs-on: ubuntu-latest

    # Steps represent a sequence of tasks that will be executed as part of the job
    steps:
      # Checks-out your repository under $GITHUB_WORKSPACE, so your job can access it

```

```

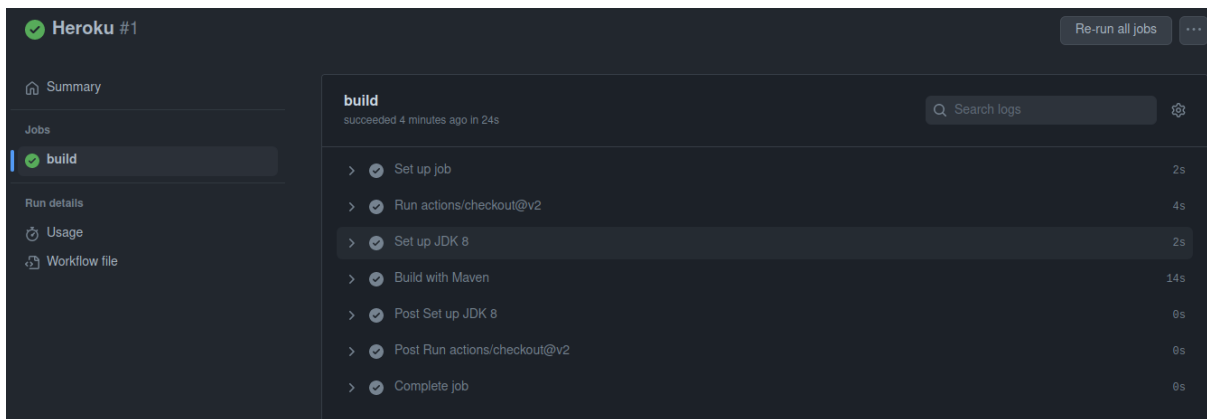
    steps:
      # Checks-out your repository under $GITHUB_WORKSPACE, so your job can access it
      - uses: actions/checkout@v2

      # Install Java JDK with maven
      - name: Set up JDK 8
        uses: actions/setup-java@v2
        with:
          heroku_api_key: ${secrets.HEROKU_API_KEY}
          heroku_app_name: "desolate-wave-51529" #Must be unique in Heroku
          heroku_email: "angelomenel@hotmail.com"
          java-version: '8'
          distribution: 'adopt'
          cache: maven

      # Compile the application
      - name: Build with Maven
        run: |
          mvn -B package --file pom.xml

```

9) Por ultimo corremos el script:



The screenshot displays the Heroku CI/CD interface for a job named "build". The job status is "succeeded 4 minutes ago in 24s". The interface includes a sidebar with navigation options: Summary, Jobs, Run details, Usage, and Workflow file. The "Jobs" section is active, showing a list of jobs with "build" selected. The main panel displays the build steps and their durations.

Step	Duration
> Set up job	2s
> Run actions/checkout@v2	4s
> Set up JDK 8	2s
> Build with Maven	14s
> Post Set up JDK 8	0s
> Post Run actions/checkout@v2	0s
> Complete job	0s