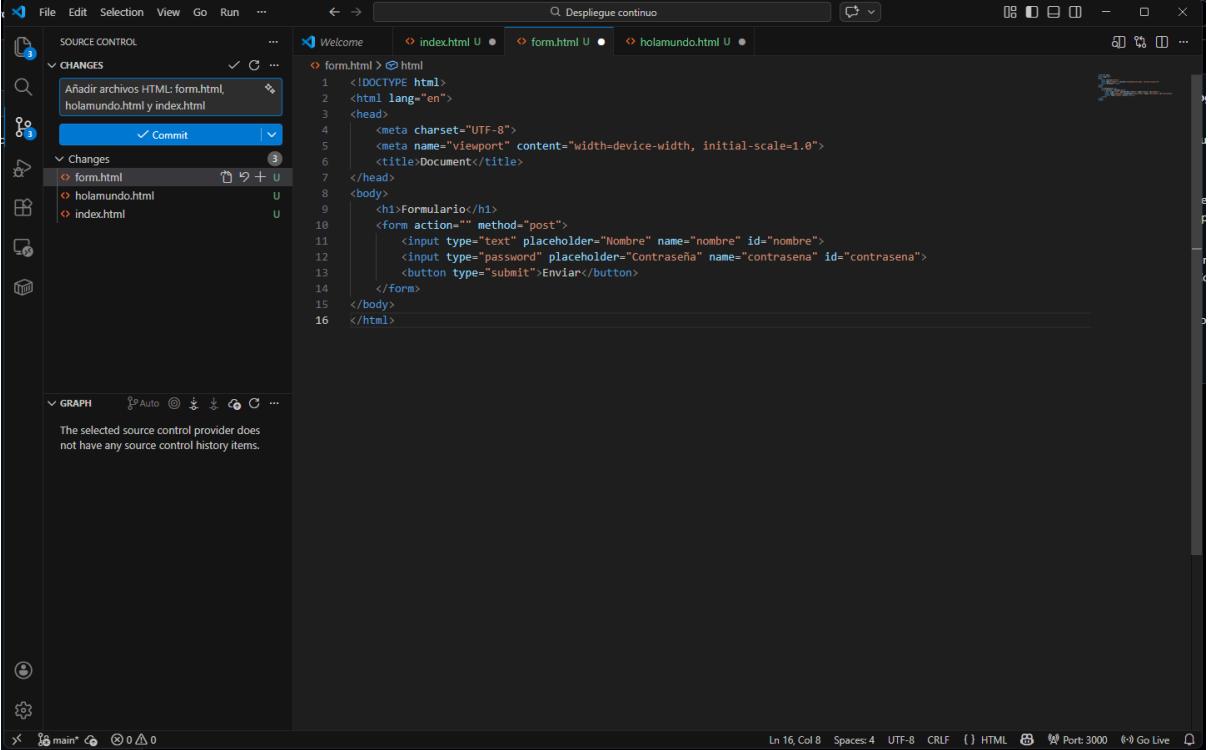


# Documentación Despliegue continuo (Github - Dockerhub)

Inicializamos el repositorio desde VSC y realizamos un commit para iniciar el git.

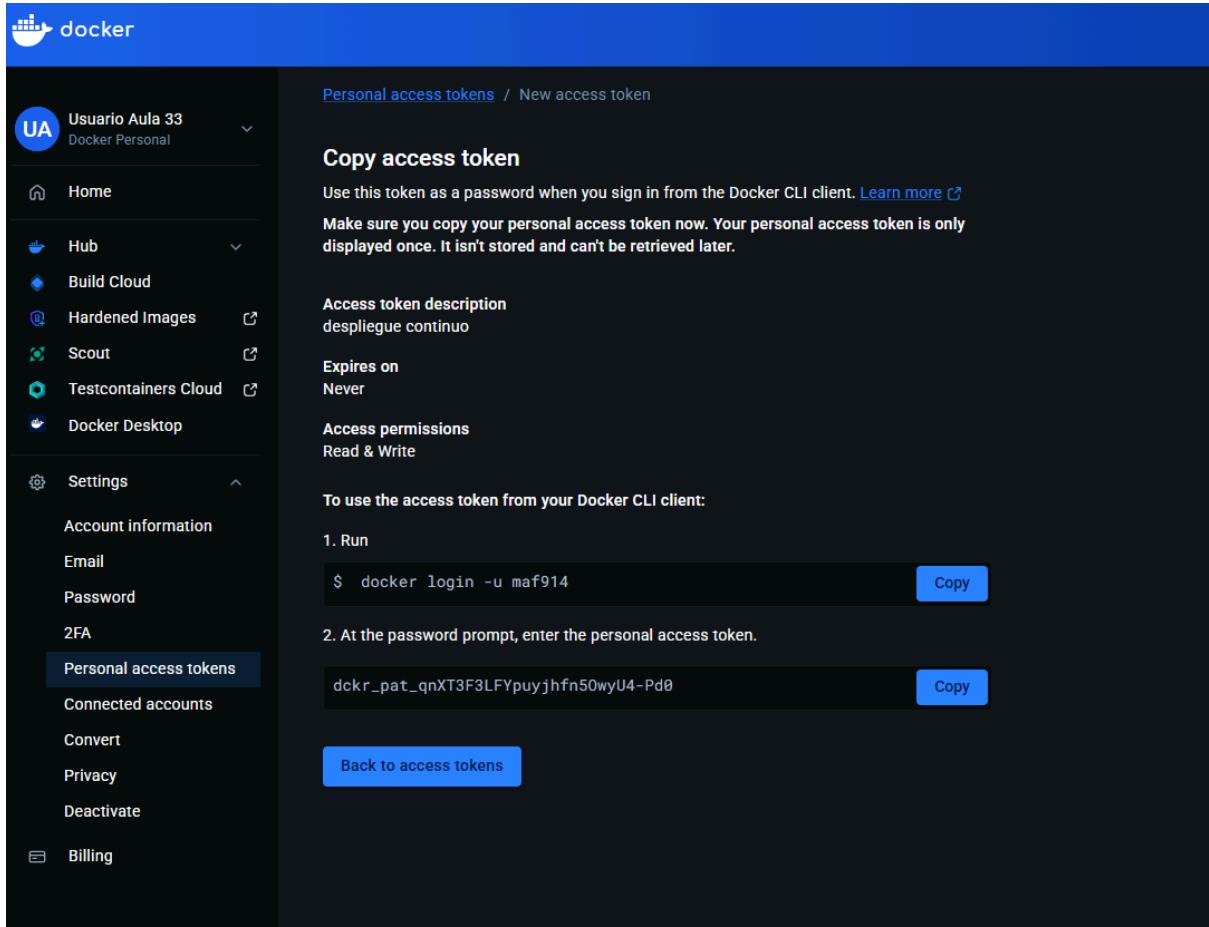


```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Documento</title>
</head>
<body>
<h1>Formulario</h1>
<form action="" method="post">
<input type="text" placeholder="Nombre" name="nombre" id="nombre">
<input type="password" placeholder="Contraseña" name="contrasena" id="contrasena">
<button type="submit">Enviar</button>
</form>
</body>
</html>
```

Creo el repositorio en GitHub y añado el git

```
PS C:\Users\dawmi\OneDrive - IES El Lago\Despliegue\Despliegue continuo> git remote add origin https://github.com/MarioAFdev/despliegue-continuo.git
PS C:\Users\dawmi\OneDrive - IES El Lago\Despliegue\Despliegue continuo> git branch -M main
PS C:\Users\dawmi\OneDrive - IES El Lago\Despliegue\Despliegue continuo> git push -u origin main
```

Creo un token de acceso personal en dockerhub para autenticarse en github.

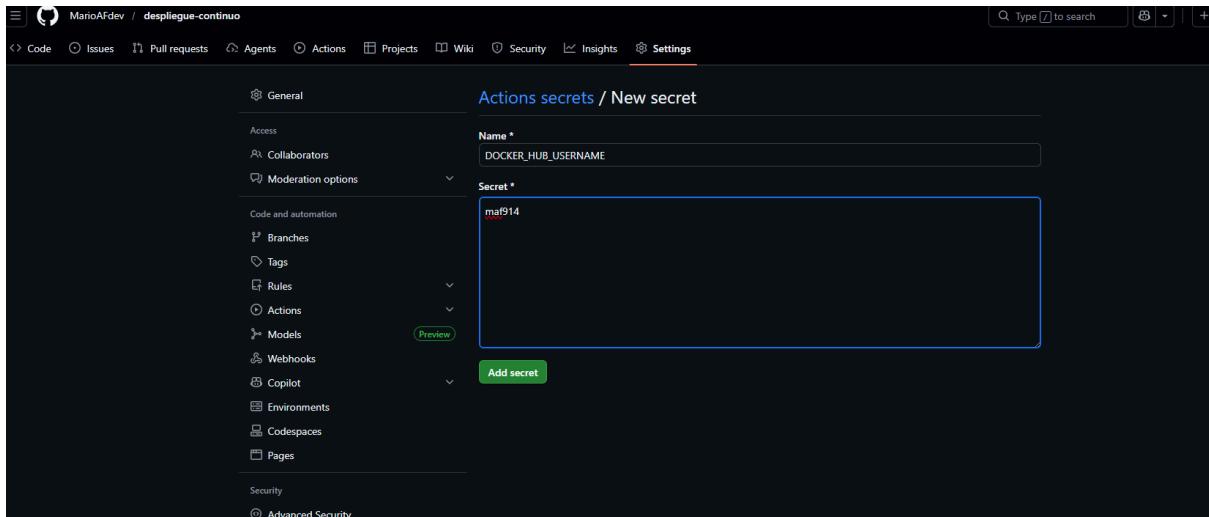


The screenshot shows the Docker Hub interface for creating a personal access token. The left sidebar shows the user profile "Usuario Aula 33" and various service links like Home, Hub, Build Cloud, etc. The "Personal access tokens" section is selected. The main content area displays instructions for using the token with the Docker CLI. It includes fields for "Access token description" (set to "despliegue continuo"), "Expires on" (set to "Never"), and "Access permissions" (set to "Read & Write"). Below these, two command examples are provided:

1. Run:  
\$ docker login -u maf914 Copy
2. At the password prompt, enter the personal access token.  
dckr\_pat\_qnXT3F3LFYpujhfn50wyU4-Pd0 Copy

A "Back to access tokens" button is at the bottom of the page.

Creo dos Secrets en el repositorio de github, uno con mi nombre de usuario de dockerhub y otro con el token de acceso que acabo de crear en dockerhub



The screenshot shows the GitHub Actions secrets creation interface. The repository path "MarioAldev / despliegue-continuo" is visible in the top bar. The "Actions" tab is selected. On the left, there's a sidebar with various GitHub features like Code, Issues, Pull requests, Agents, Actions, Projects, Wiki, Security, Insights, and Settings. The "Actions" section is expanded, showing sub-options like General, Access, Collaborators, and Moderation options. The main content area is titled "Actions secrets / New secret". It has fields for "Name" (set to "DOCKER\_HUB\_USERNAME") and "Secret" (containing the value "maf914"). A "Preview" button is visible next to the secret input field, and a "Add secret" button is at the bottom right.

The screenshot shows the GitHub repository settings page for 'despliegue-continuo'. The left sidebar has sections like General, Access, Collaborators, Moderation options, Code and automation (Branches, Tags, Actions, Models, Webhooks, Copilot, Environments, Codespaces, Pages), Security (Advanced Security, Deploy keys, Secrets and variables), and Actions. The 'Actions' section is currently selected. The main area is titled 'Actions secrets and variables'. It explains that secrets and variables manage reusable configuration data. It shows two environment secrets: DOCKERHUB\_TOKEN (created now) and DOCKER\_HUB\_USERNAME (created 2 minutes ago). A green button 'New repository secret' is visible.

## Creo un workflow de github actions (.yml)

The screenshot shows the GitHub Actions editor for the 'main.yml' workflow file. The code is as follows:

```

1 # Nombre de nuestro flujo de trabajo
2 name: Construir y Publicar en Docker Hub
3
4 # Cuando se ejecuta?
5 on:
6   push:
7     branches: [ "main" ] # Se ejecuta al hacer push a la rama 'main'
8   pull_request:
9     branches: [ "main" ] # También al crear un PR hacia 'main'
10
11 # Un workflow se compone de uno o más jobs
12 jobs:
13   # Este job se llama "build-and-push"
14   build-and-push:
15     # build-and-push:
16       runs-on: ubuntu-latest # El sistema operativo de la máquina virtual donde correrá
17
18       # Los pasos que ejecutará el job
19       steps:
20         # 1. Descargar el código del repositorio
21         - name: Checkout del repositorio
22           uses: actions/checkout@v4
23
24         # 2. Iniciar sesión en Docker Hub usando nuestros secretos
25         - name: Iniciar sesión en Docker Hub
26           user: docker/login-action@v3
27           with:
28             username: ${{ secrets.DOCKER_HUB_USERNAME }}
29             password: ${{ secrets.DOCKER_HUB_ACCESS_TOKEN }}
30
31         # 3. Configurar Docker Buildx (para construir imágenes más rápido y con más opciones)
32         - name: Configurar Docker Buildx
33           uses: docker/setup-buildx-action@v3
34
35         # 4. Construir y publicar la imagen
36         - name: Construir y publicar imagen
37           uses: docker/build-push-action@v5

```

The right side of the editor shows the Marketplace for Actions with various popular actions listed. A green 'Commit changes' button is at the top right.

## Como no tengo Dockerfile no cree bien el workflow

The screenshot shows the GitHub Actions workflow details for a repository named 'despliegue-continuo'. The workflow is triggered via push and has one job named 'build-and-push'. The status of the job is 'Failure' with a duration of 17s. The error message in the annotations section states: 'buildx failed with: ERROR: failed to build: failed to solve: failed to read dockerfile: open Dockerfile: no such file or directory'.

## Añado el dockerfile y corrijo el workflow

The screenshot shows the GitHub Actions workflow runs for the same repository. The workflow now has two successful runs, both triggered by 'workFlow\_dispatch' event triggers. The first run was 3 minutes ago and the second was 36 minutes ago. Both runs were completed 28s ago. The status is 'Success' for both runs.

## Compruebo que se haya creado la imagen en mi repositorio de dockerhub

The screenshot shows the Docker Hub interface for the repository `maf914/despliegue-continuo-miweb`. The left sidebar shows the user's profile and navigation links for repositories, collaborations, settings, billing, usage, pulls, and storage. The main content area shows the repository details, including the name `maf914/despliegue-continuo-miweb`, a note about it being a Hardened Image, and a link to its general documentation. It also shows the repository size as 24.8 MB and the last push time as 6 minutes ago. There are sections for adding a description, category, and Docker commands (with a command line example: `docker push maf914/despliegue-continuo-miweb:tagname`). A sidebar for `buildcloud` offers to build with Docker Build Cloud, mentioning faster builds through shared caching and native multi-platform support.

Tag	OS	Type	Pulled	Pushed
3af55b40a2d4bedbf8...		Image	less than 1 day	6 minutes
latest		Image	less than 1 day	6 minutes