

**Alquipiso  
Grupo 3.8**

**DESPLIEGUE DOCKER**

Debido a que una imagen de Docker puede no funcionar en otros equipos si las arquitecturas de los sistemas son diferentes (ya que la imagen incluye software y binarios compilados para la arquitectura donde se creó), vamos a explicar dos formas de desplegar la aplicación.

Hemos entregado dos binarios de la imagen uno compatible con arquitectura arm(Silicon) y otro con amd(Ubuntu).

**Prerequisitos:**

* Debemos tener instalado Docker Desktop en nuestro equipo. Lo podemos hacer desde su [página web oficial](https://www.docker.com/products/docker-desktop/) (<https://www.docker.com/products/docker-desktop/>).

Una vez instalado ejecutamos docker –versión y deberíamos ver la versión de Docker.

* Tener descargados los binarios por ejemplo en la carpeta Descargas de nuestro equipo.

**Pasos a seguir usando los binarios proporcionados**

1. Iniciamos Docker abriendo la aplicación.
2. Abrimos un terminal en la carpeta Descargas (o donde se encuentren los binarios)
3. Ejecutamos docker load < nombre\_del\_binario.tar
4. Ejecutamos docker run -p 8000:8000 alquipiso
5. Accedemos a la aplicación web desde: <http://localhost:8000/>

**Pasos a seguir si los binarios no fueran compatibles**

Si los binarios no son compatibles, tendremos que clonar el repositorio de la aplicación y construir la imagen a partir del Dockerfile.

1. Abrimos una terminal y clonamos el repositorio: git clone <https://github.com/alenicbra/alquipiso.git>
2. Entramos a la carpeta: cd alquipiso
3. Construimos la imagen: docker build -t alquipiso . (importante el “.” al final)
4. Arrancamos la imagen: docker run -p 8000:8000 alquipiso
5. Accedemos a la aplicación web desde <http://localhost:8000/>

**Consideraciones:**Para hacer login:

* Como cliente: [cliente1@example.com](mailto:cliente1@example.com) / cliente1
* Como propietario [propietario1@alquipiso.com](mailto:propietario1@alquipiso.com) / propietario1

¡Importante!

Nuestra aplicación está desplegada en la nube y es accesible desde <https://alenicbra.pythonanywhere.com/>. Es importante esto para la comprobación de algunas funcionalidades. Nuestro equipo para validar las reservas ha usado un webhook de nuestra pasarela de pago (Stripe) que nos facilita información sobre el pago de las reservas (una reserva no se confirma hasta que no esté pagada). Es decir, para que una reserva salga como pagada habría que realizarla desde ese endpoint. Además, tanto la notificación interna de confirmación de pago como el envío del correo de confirmación al cliente depende también de ese webhook.