



Tecnológico de Monterrey

Campus Querétaro

Manual Prueba de Arquitectura

Gamaliel Marines Olvera	A01708746
Uri Jared Gopar Morales	A01709413
José Antonio Miranda Baños	A01611795
María Fernanda Moreno Gómez	A01708653
Oskar Adolfo Villa López	A01275287
Luis Ángel Cruz García	A01736345

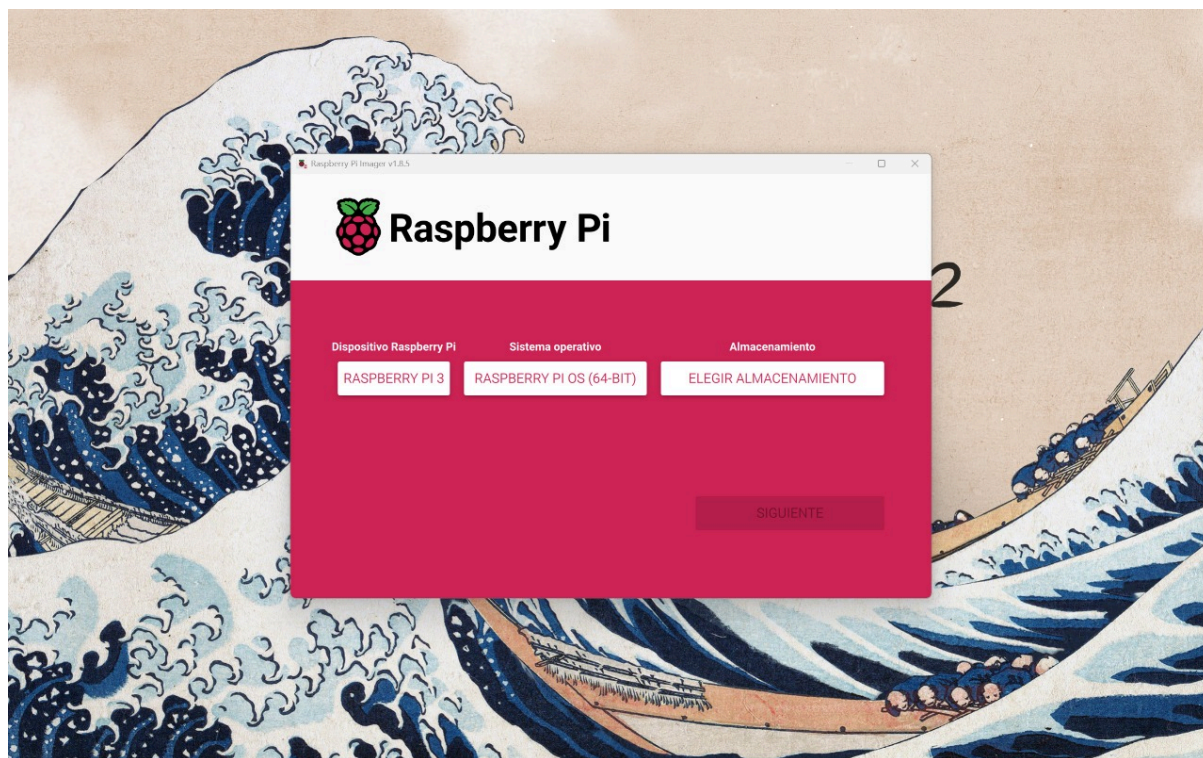
Inteligencia artificial avanzada para la ciencia de datos II
Grupo 501

Introducción

Este documento detalla el desarrollo mediante un manual, los pasos a seguir para realizar correctamente la validación de nuestra prueba de arquitectura; en esta prueba, se modeló la infraestructura de nuestro socio formador (el CAETEC) y se ejecutó nuestro modelo en dicha arquitectura.

Pasos a realizar:

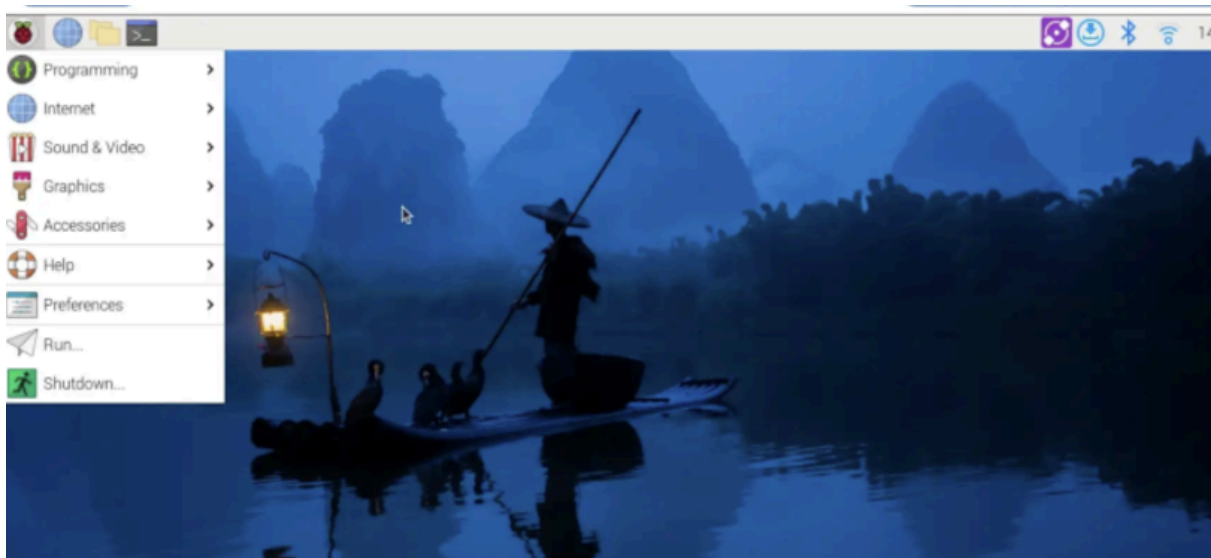
1. Descargar el sistema operativo de [Raspberry Pi](#), debe de ser un sistema de 64 bits, esto es porque para las tecnologías utilizadas no soporta una versión de 32 bits. En nuestro caso ocupamos Raspbian de 64 bits en una Raspberry Pi Model 3 2015, este sistema operativo se debe de guardar en la MicroSD.



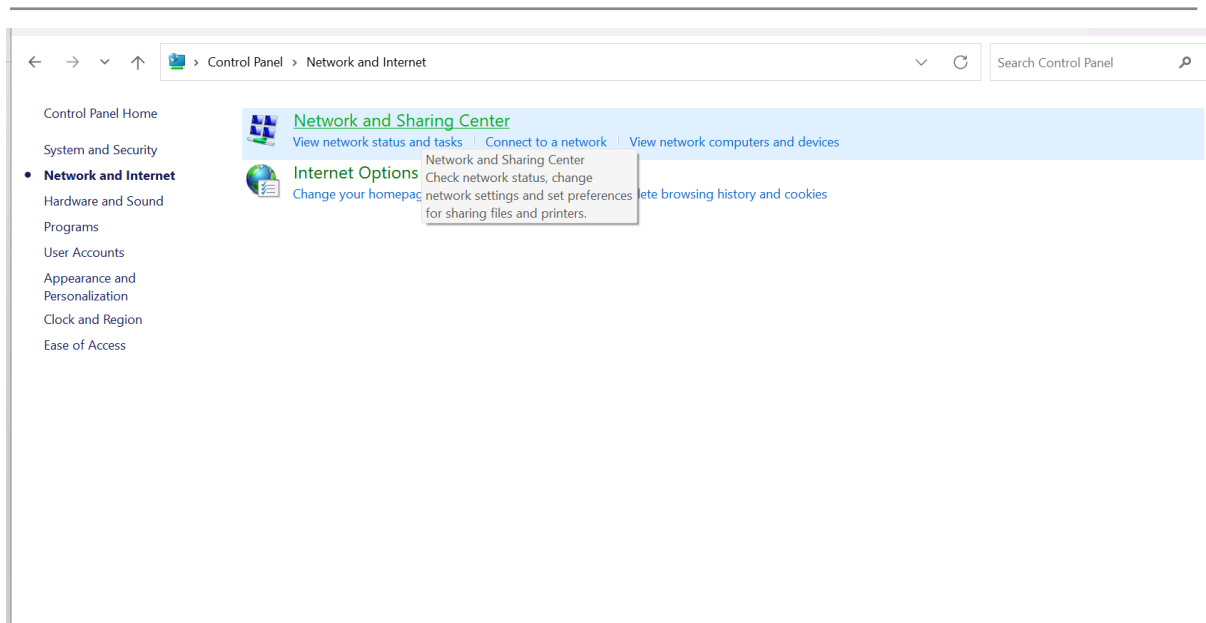
2. Una vez con nuestro sistema operativo descargado, colocamos la MicroSD en la ranura con la que cuenta la raspberry, también conectamos el monitor, teclado y mouse, realizadas las conexiones seremos capaces de conectar a la luz, esto lo realizamos porque el monitor nos ayudará para poder visualizar cuando nuestro sistema operativo inicialice.
3. Al inicializar el sistema operativo se nos mostrará una pantalla donde nos pedirá que creamos un usuario con su contraseña. Este usuario y contraseña nos servirá para poder establecer la conexión ssh más adelante (No olvidar).



4. Sabremos que inicializó correctamente cuando cargue la pantalla de inicio del home, como una computadora normal.



5. Como siguiente paso debemos pasar internet a nuestra raspberry, por lo que en nuestra laptop conectaremos el cable ethernet a nuestra raspberry, para configurar esta opción desde nuestra laptop tenemos que seguir los siguientes pasos:
 - a. Entrar al panel de control -> Network y Internet -> Network and Sharing Center.




- b. Al entrar a dicha ventana le daremos clic en nuestra red conectada, en este caso es Wi-Fi (Tec)

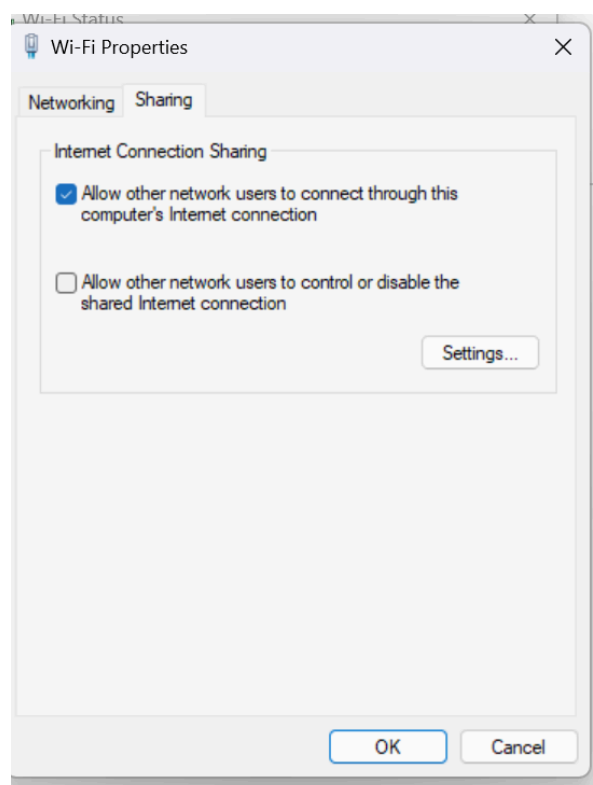
Tec

Public network

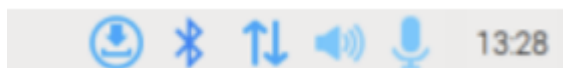
Access type: Internet

Connections:  Wi-Fi (Tec)

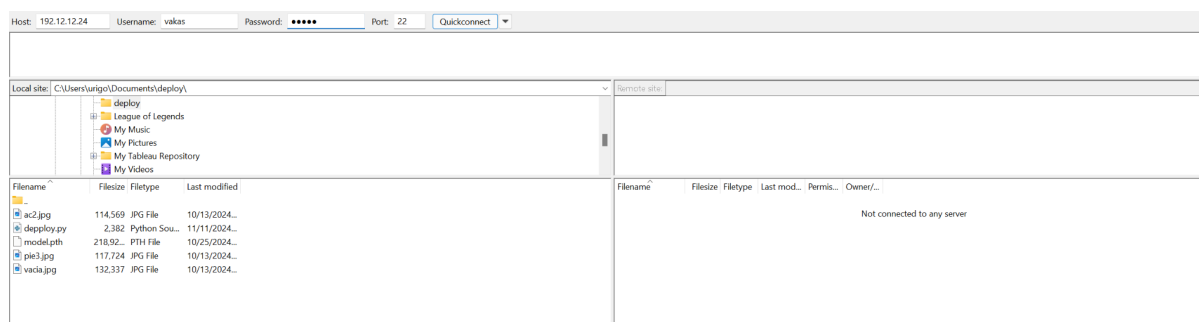
- c. Entraremos en Properties -> seleccionamos la pestaña de Sharing y marcamos la primera casilla, al final le damos OK para que se guarde la configuración.



6. Para saber si nuestra raspberry ya cuenta con conexión a Internet debemos de visualizar el icono de flechas en color azul.



7. Al pasar el mouse por este icono nos mostrará la dirección IP asignada, no se mostrará dicha IP por seguridad.
8. Gracias a la herramienta de [FileZilla](#) podremos conectarnos desde nuestra computadora a la Raspberry esto para pasar los documentos necesarios para poder correr el modelo.
9. FileZilla se nos abrirá de la siguiente manera, como podemos apreciar en Host pondremos la dirección IP de nuestra Raspberry, en username y password usaremos las mismas credenciales que escribimos al inicializar el dispositivo. Por último, en el puerto 22, este puerto es el general para conexiones de SSH.



10. Los archivos que debemos seleccionar se encuentran en el siguiente link, en este se encuentran, el script con el que se corre las predicciones e imágenes de prueba para ver el desempeño de predicciones.:

<https://github.com/Mirabay/Vakas/tree/main/deploy-%20architecture>

11. En la terminal de Raspberry Pi tenemos que instalar Pytorch para esto es necesario descargar miniconda y crear un ambiente virtual el cual nos ayudará para descargar una versión más compacta de Pytorch. Si por alguna razón no pueden descargar miniconda les recomiendo el siguiente tutorial:
- a. [Tutorial de Miniconda en una Raspberry](#)
12. Una vez seguido el tutorial y tener el ambiente virtual, nos tenemos que dirigir a la carpeta donde guardamos el script descargado y corremos el archivo de deploy.py. NOTA: Para que se ejecute las predicciones debemos de tener el

modelo preentrenado, sin embargo, en el repositorio no se encuentra por términos de seguridad.

13. Al ejecutar dicho script nos tiene que imprimir lo siguiente, donde podremos visualizar cuanto se tardó en cargar el modelo y con se tardó en realizar las predicciones:

```
quit()
(vakas) vakas@raspberrypi:~ $ cd Documents
(vakas) vakas@raspberrypi:~/Documents $ cd vakas
-bash: cd: vakas: No such file or directory
(vakas) vakas@raspberrypi:~/Documents $ cd Vakas
(vakas) vakas@raspberrypi:~/Documents/Vakas $ ls
x2.jpg  deploy.py  model.pth  pie3.jpg  vacia.jpg
(vakas) vakas@raspberrypi:~/Documents/Vakas $ python deploy.py
/home/vakas/Documents/Vakas/deploy.py:61: FutureWarning: You are using 'torch.load' with 'weights_only=False' (the current default value), which uses the default pickle module implicitly. It is possible to construct malicious pickle data which will execute arbitrary code during unpickling (See https://github.com/pytorch/pytorch/blob/main/SECURITY.md#untrusted-models for more details). In a future release, the default value for 'weights_only' will be flipped to 'True'. This limits the functions that could be executed during unpickling. Arbitrary objects will no longer be allowed to be loaded via this mode unless they are explicitly allowlisted by the user via 'torch.serialization.add_safe_globals'. We recommend you start setting 'weights_only=True' for any use case where you don't have full control of the loaded file. Please open an issue on GitHub for any issues related to this experimental feature.
  model.load_state_dict(torch.load(model_path, map_location=device))
Predicción: cana_vacia
Predicción: vaca_acostada
Tiempo de cargar el modelo: 29.1779 segundos
Tiempo de ejecución: 6.4226 segundos
(vakas) vakas@raspberrypi:~/Documents/Vakas $ |
```