# Document My Experience

The exercise assigned to me has been a unique experience. It enabled me to step into the world of Dockers, Though I have been working on Cloud-native projects, I did not have a direct experience of deployment. What was more gratifying was that I had the hands-on experience of understand dockers, installing the Docker desktop, and through the Windows platform and the PowerShell as the interface, I was able to develop a sample application with the help of the instructions. Let me elaborate the activities that I performed in each step:   
  
Step 1:     Install Docker Desktop on Windows

Docker Desktop on Windows enables to build and share containerized applications and microservices.   
As a preparatory step, I had checked the System Requirements, and enabled Windows Subsystem for Linux 2 on Windows machines, Docker Desktop provided the ability to work natively on Linux through WSL2 on Windows machines.  
I would like to present a screen shot that was captured during the process of Docker Desktop for Windows installation.

Graphical user interface, text, application

Description automatically generated

Fig 1: Docker Desktop installation  
  
Fig 2: Docker Desktop

Features:  
Docker Desktop provide the ability to toggle between Linux and Windows Server environments to build applications.

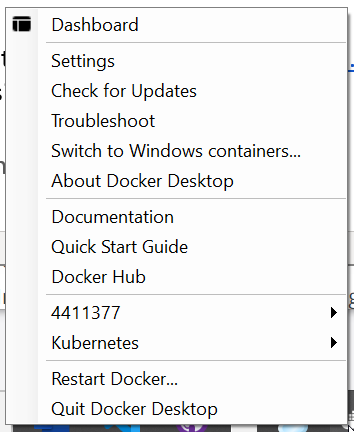


Fig 3: Docker Desktop icon right-click  
  
Verification:  
The Docker version command was checked:  
Text

Description automatically generated  
Fig 4: Docker Version  
  
Once installed it enabled to:  
a. Clone a GIT repository to the local docker image

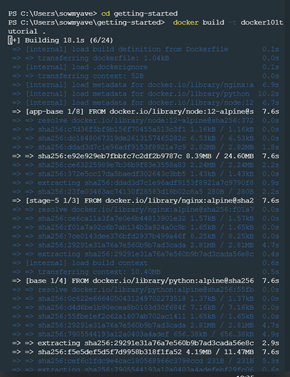
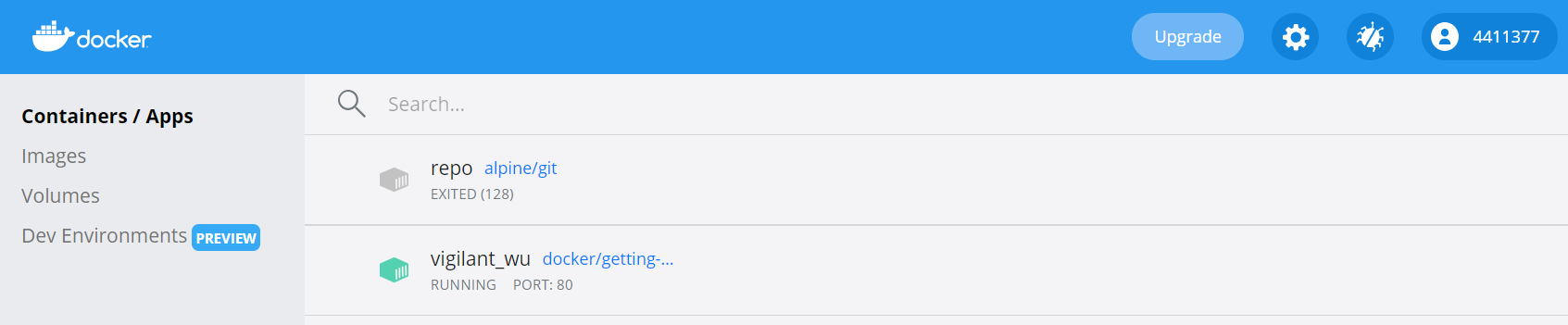
b. Build  
c. Run

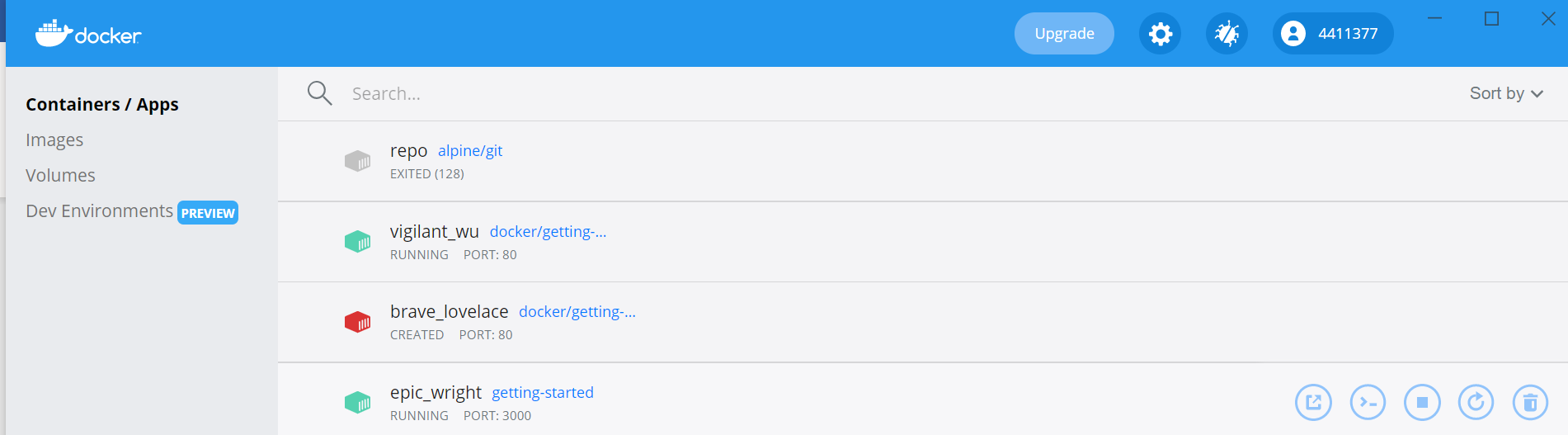
d. Share (optional step)  
For example:  
Text

Description automatically generated  
What went well  
The instructions that were part of <https://docs.docker.com/docker-for-windows/install/> are clear and comprehensive.

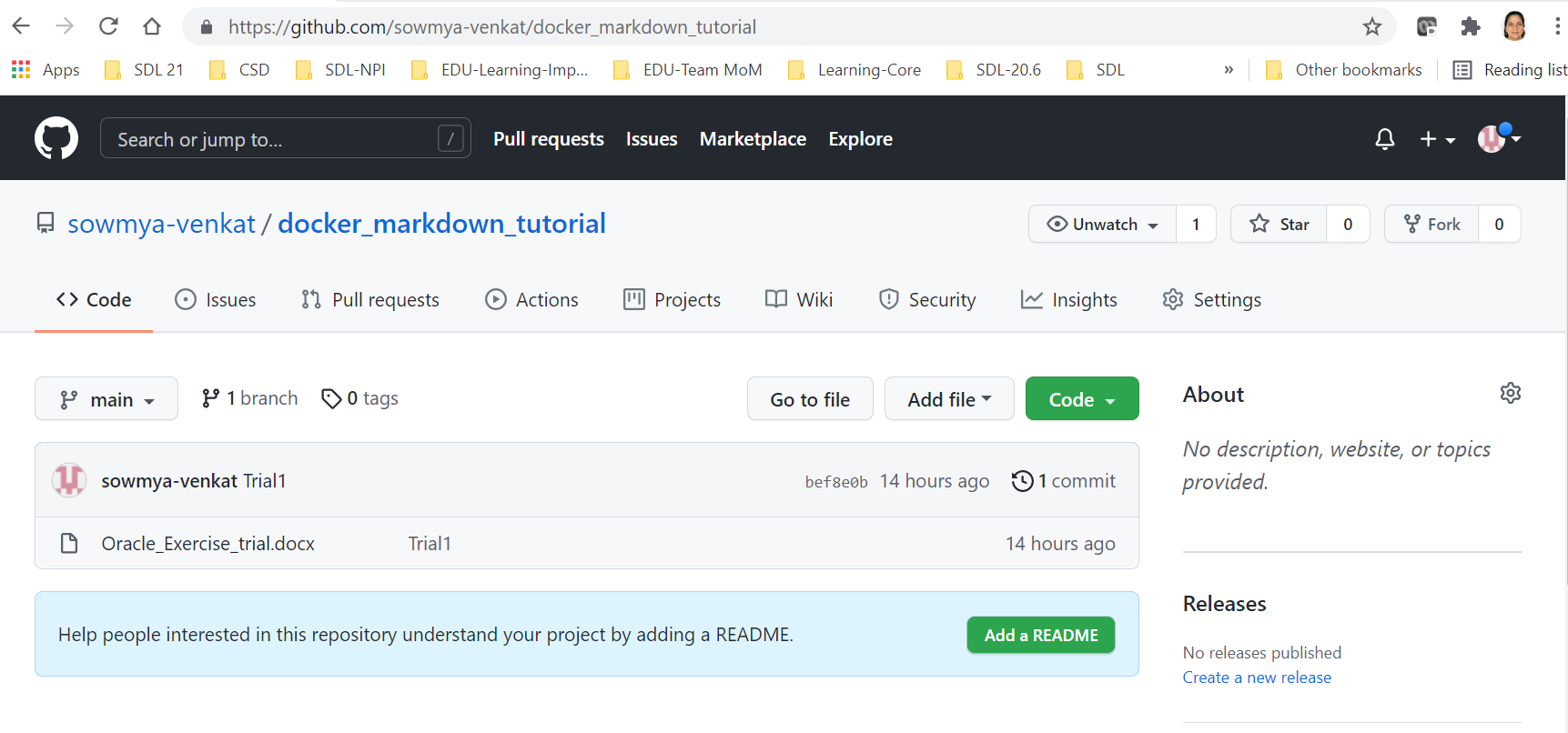
What needs improvement  
  
The **System Requirements** section has information covered in two tabs, namely **WSL2 Background** and **Hyper-V backend and Windows containers**. It is not clearly mentioned if both the requirements must be essentially satisfied.

Step 2: Get familiar with Docker Desktop  
  
In this step, I got familiar with the use of the Docker Run command and when run successfully, the new Container information appears in the Docker Desktop

  
Fig 5: Docker command on the command prompt  
  
  
Fig 6: Docker Desktop with the new app Container in the running state

Step 3: Download and Run a Simple Application in a Docker Container  
  
During this exercise, I learnt how to build a container image with the help of a Dockerfile. The aim of this exercise is to build a simple todo list manager that runs in Node.js.  
When an image was built, the container had started and the app was in the running state.   
The following figure indicates that 2 Docker containers are currently running in two different ports:  
 a. vigilant\_wu in Port 80 and   
 b. epic\_wrightin port 3000 (newly created container image)  
  
  
Fig 7: App container images   
When it was tested in the browser (<http://localhost:3000>), the todo list manager running in Node.js appeared as follows:  
Graphical user interface, text, application, email

Description automatically generated  
  
Step 4: Publishing in the GitHub  
As a part of this exercise, I have created an GitHub account

  
  
I have been fascinated with working on the practical side of development.