Prototype Development

PBT205—Project-based Learning Studio: Technology

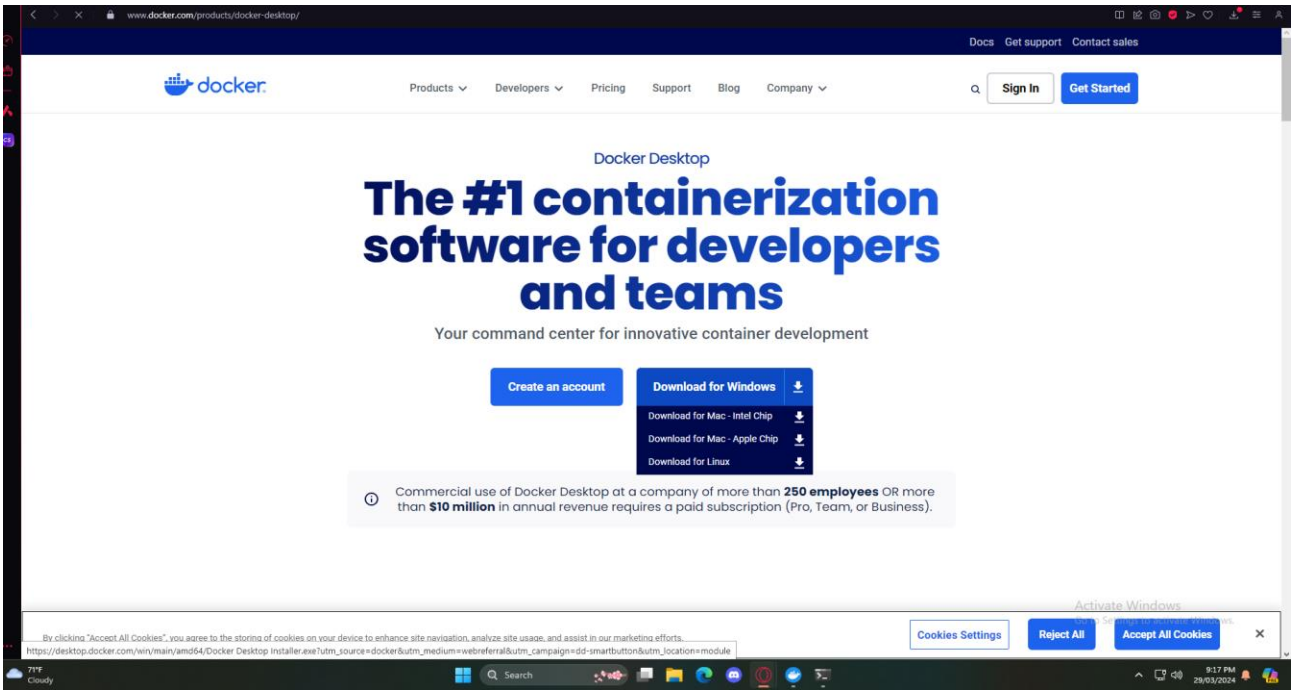
Name : Quan Thang Tran

Student ID: A 00102814

Lecturer : RUBINA SARKI

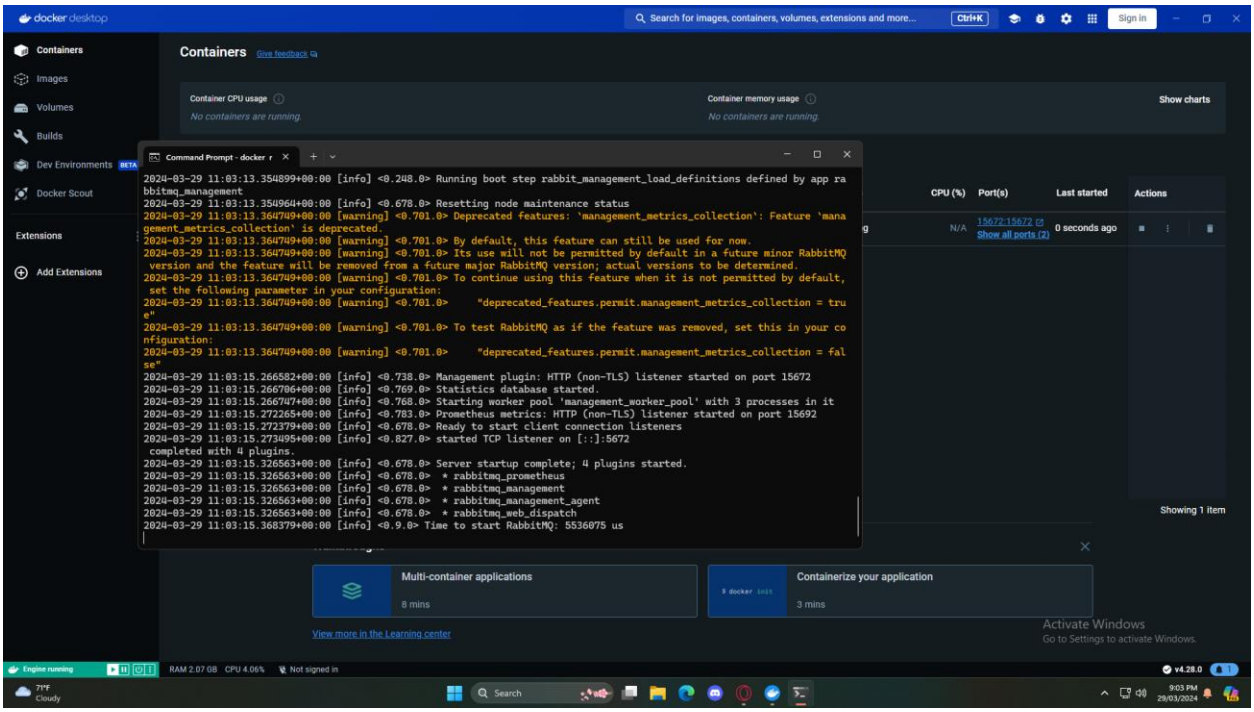
First of all,
As I begin to design an application for a trading system, I see how crucial it is to set up middleware to enable smooth communication between components. My preferred middleware option is RabbitMQ since it is known for being dependable in message queuing. I'll outline the exact steps I took to set up RabbitMQ using Docker in this report so that I may more easily construct my trading system application.

Step 1: Installing and configuring Docker
I'll start by making sure Docker is installed on my local computer to get things started. Application deployment is made easier by Docker's containerization technology, which also guarantees consistency between environments. I'll go to the official Docker website and download the OS-specific version of Docker Desktop. Docker Desktop offers an easy-to-use interface for controlling containers after it is installed. The groundwork for smoothly operating RabbitMQ instances is laid in this stage.



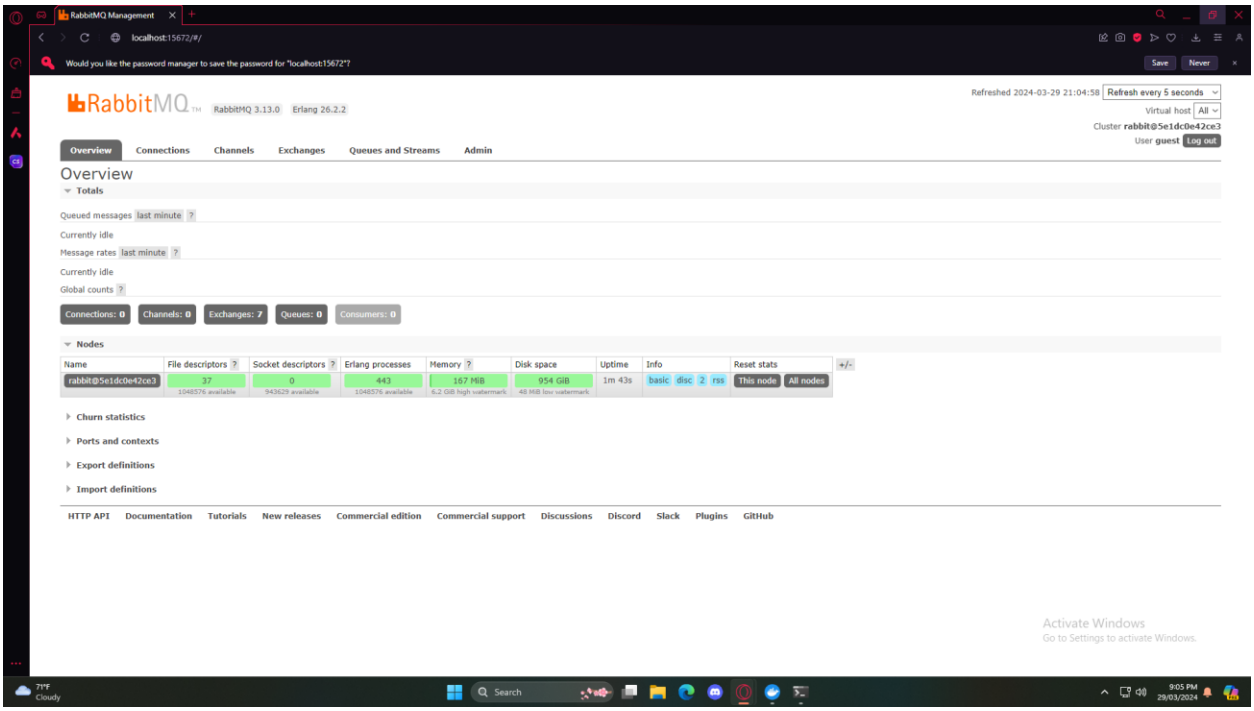
Step 2: Operating the RabbitMQ Container After setting up Docker, I typed the following command into my terminal: docker run -it --rm --name rabbitmq -p 5672:5672 -p 15672:15672 rabbitmq:3-management. This command started

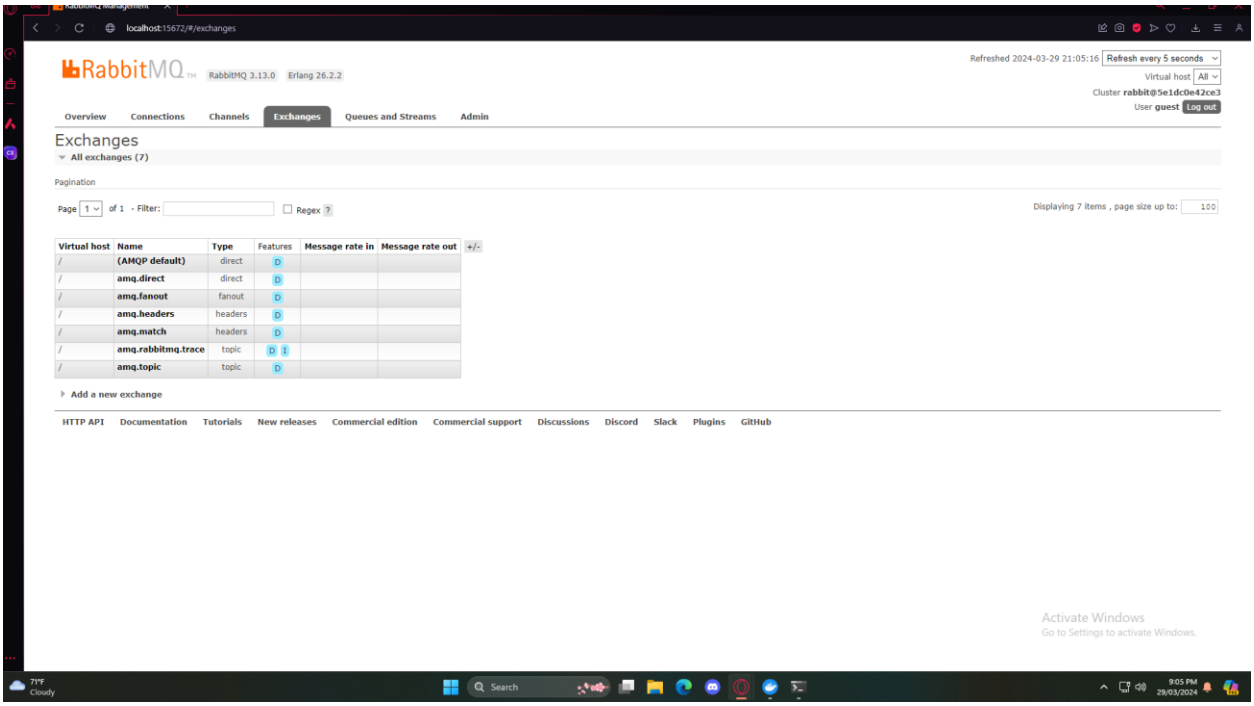
a container called "rabbitmq" with port mappings for AMQP and the administration UI in addition to fetching the RabbitMQ image.



Step 3: Use the optional RabbitMQ Management Interface

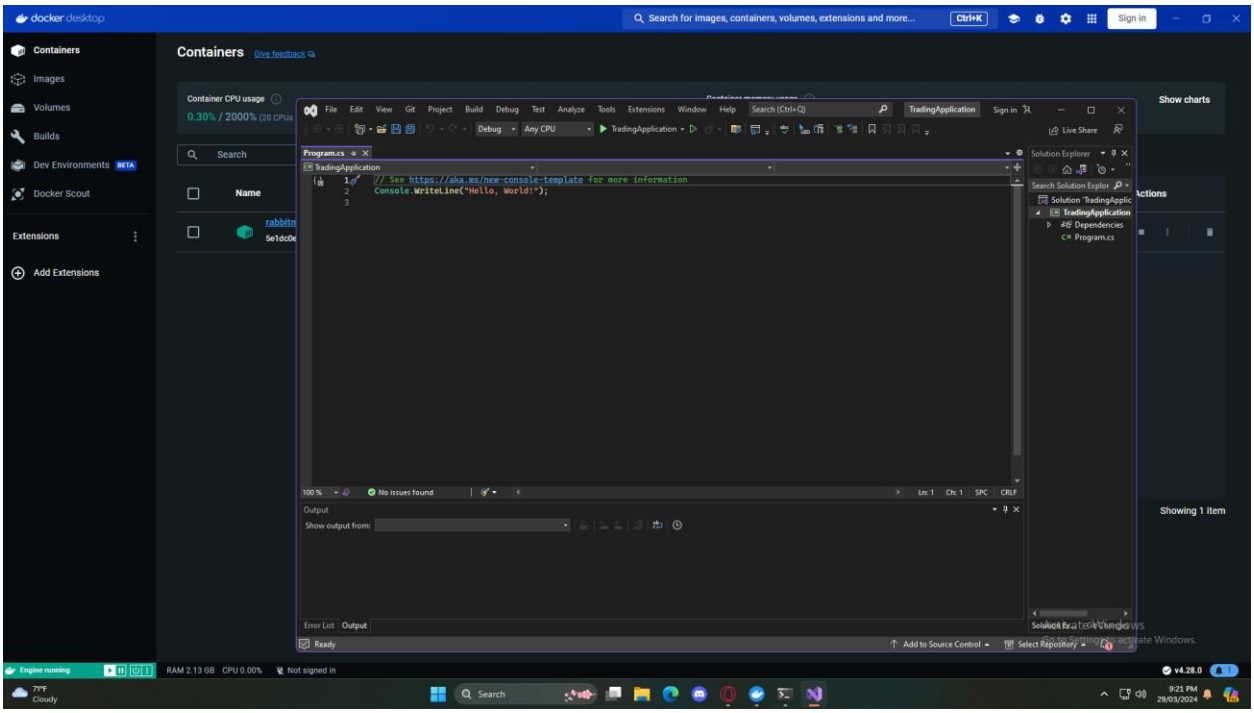
I opened my web browser and navigated to http://localhost:15672 to view the RabbitMQ management interface. This interface offered priceless insights on RabbitMQ-related exchanges, queues, and other data. This was an optional step, but it worked well for debugging and monitoring.





Final Action

Once my computer has completed launching Docker and Rabbitmq, I'll begin to Visual Studio Code and kick off coding.


docker desktop
Search for images, containers, volumes, extensions and more...
Ctrl+K
Sign in
Containers
Images
Volumes
Builds
Dev Environments
Docker Scout
Extensions
Add Extensions
Containers
Give feedback
Container CPU usage
0.30% / 2000%
Search
Name
Show charts
Showing 1 item
Program.cs
TradingApplication
// See https://aka.ms/new-console-template for more information
Console.WriteLine("Hello, World!");
No issues found
Output
Show output from:
Error List
Output
Ready
Add to Source Control
Select Repository
Solution Explorer
Dependencies
Program.cs
Live Share
Debug
Any CPU
Engine running
RAM 2.13 GB
CPU 0.00%
Not signed in
v4.28.0
71°F
Cloudy
Search
9:21 PM
29/03/2024