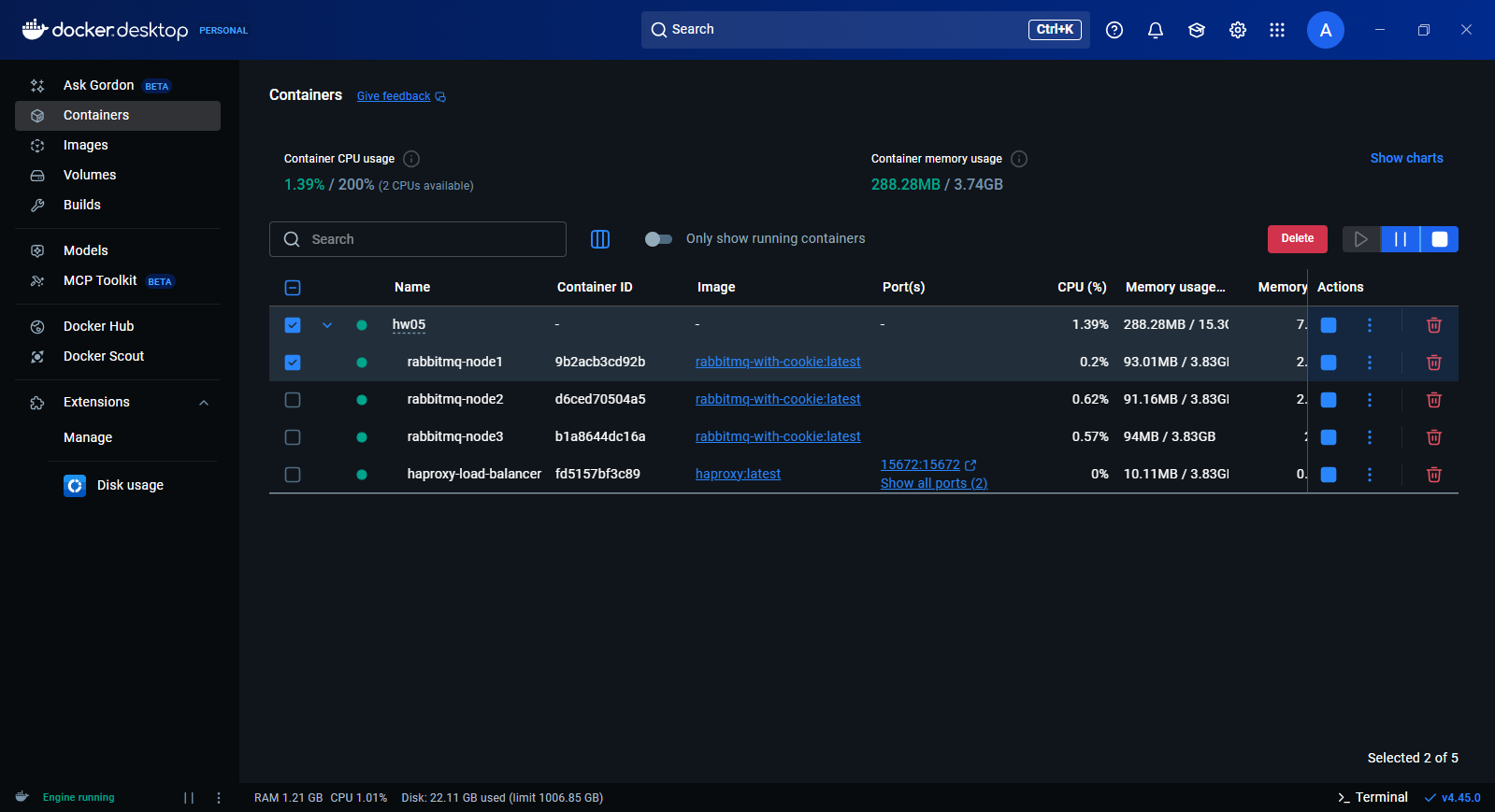
# 2025-07-OTUS-RABBITMQ

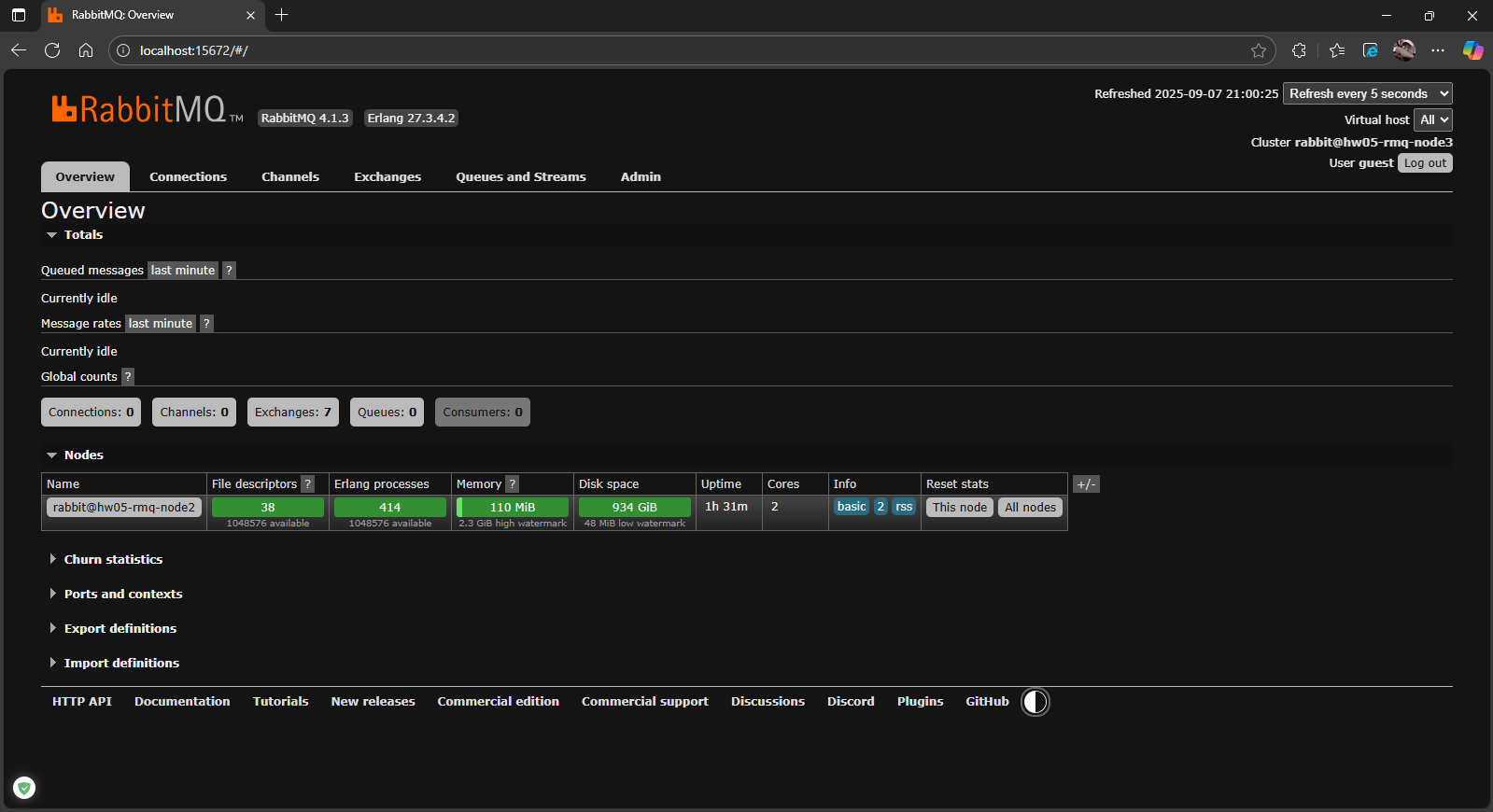
## Домашнее задание №5

## Александр Николаев

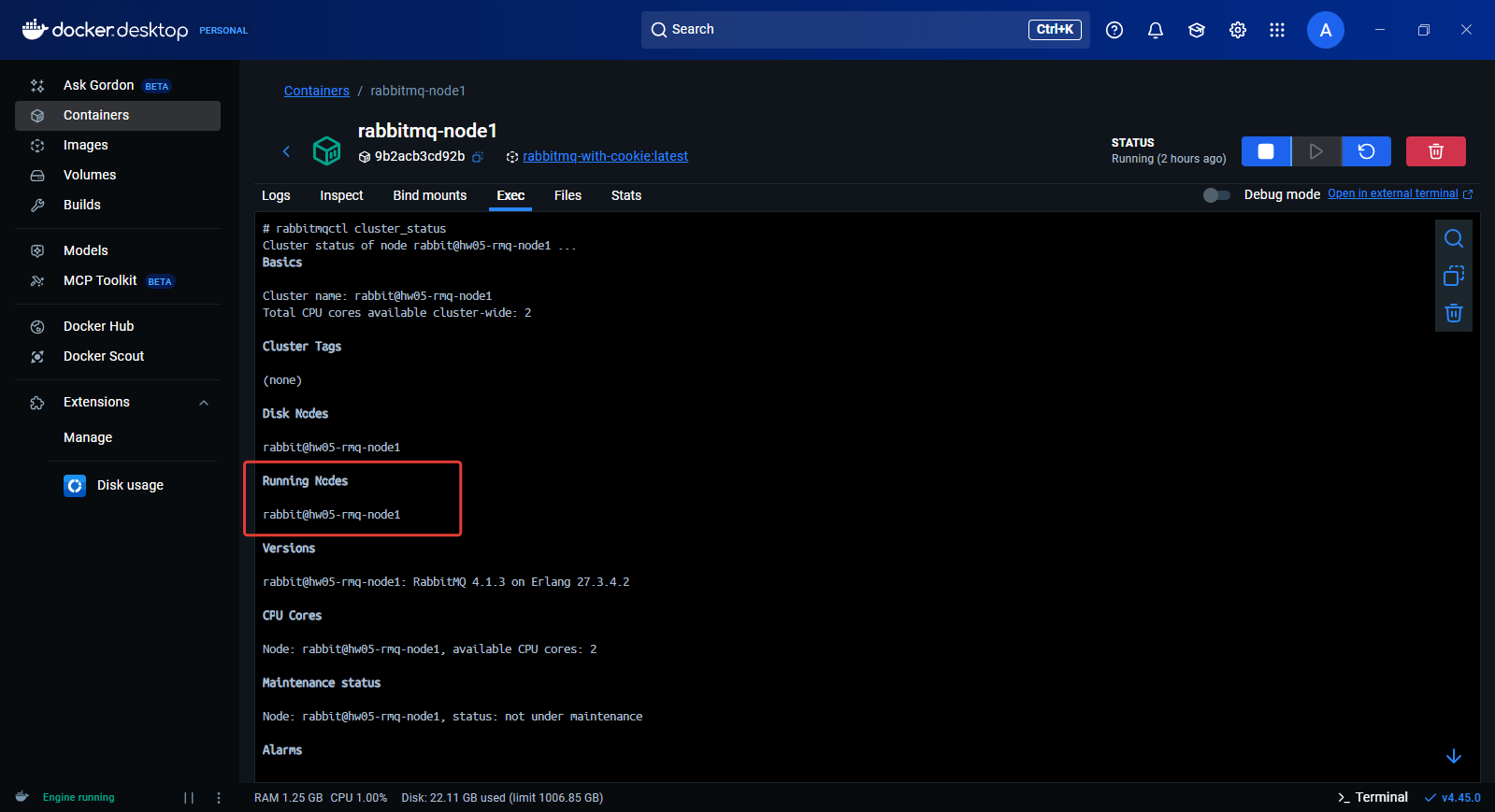
1. Создал Dockerfile для кастомизации стандартного image RabbitMQ. Заранее скопировал файл .erlang.cookie в свой image.
2. Развернул 3 независимых ноды RabbitMQ в docker с помощью docker-compose.yaml



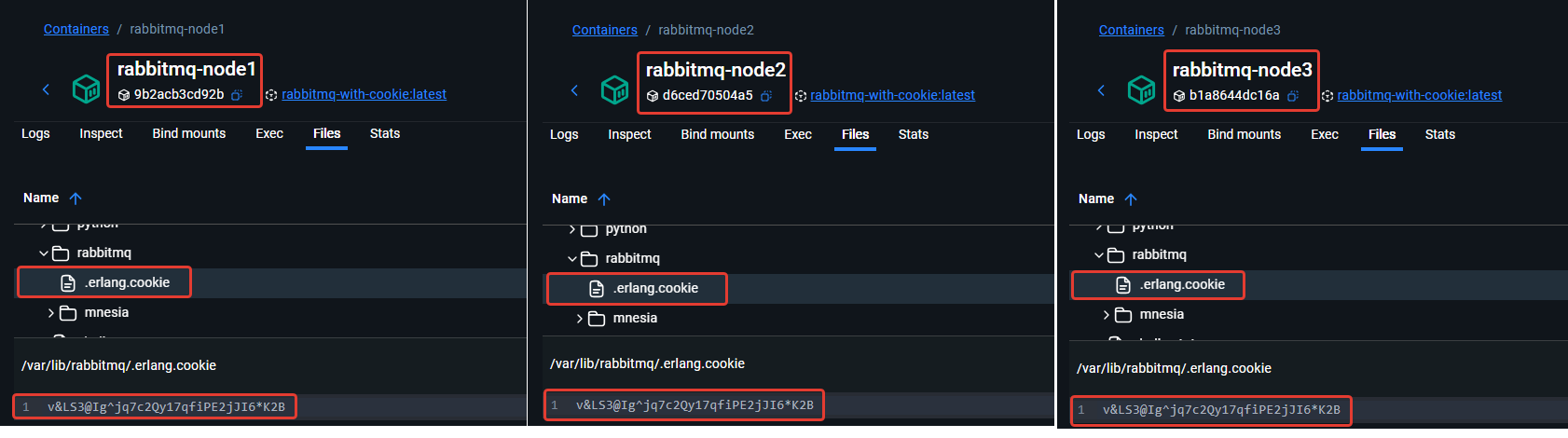
1. Проверил, что management console RabbitMQ запускается в браузере по адресу: <http://localhost:15672/>.



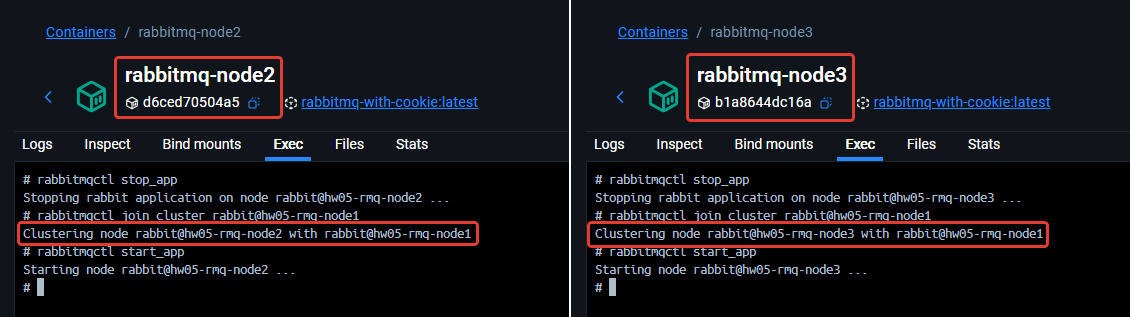
1. Убедился в том, что кластер не собран и состоит из одной ноды. Запустил команду “**rabbitmqctl cluster\_status**” на ноде, которая в дальнейшем будет выступать в качестве master ноды кластера.



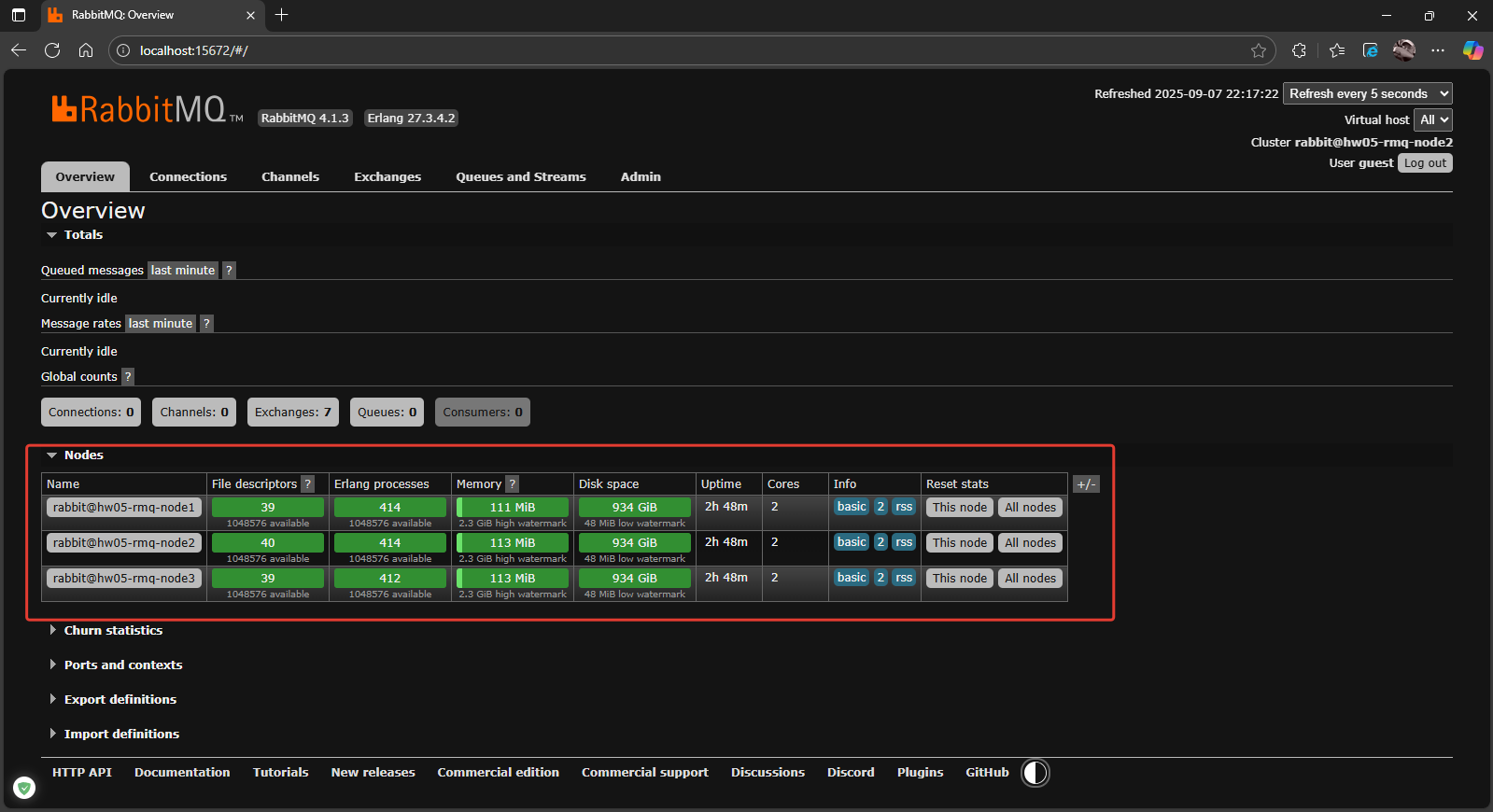
1. Убедился, что файл .erlang.cookie есть на каждой ноде и он одинаковый.



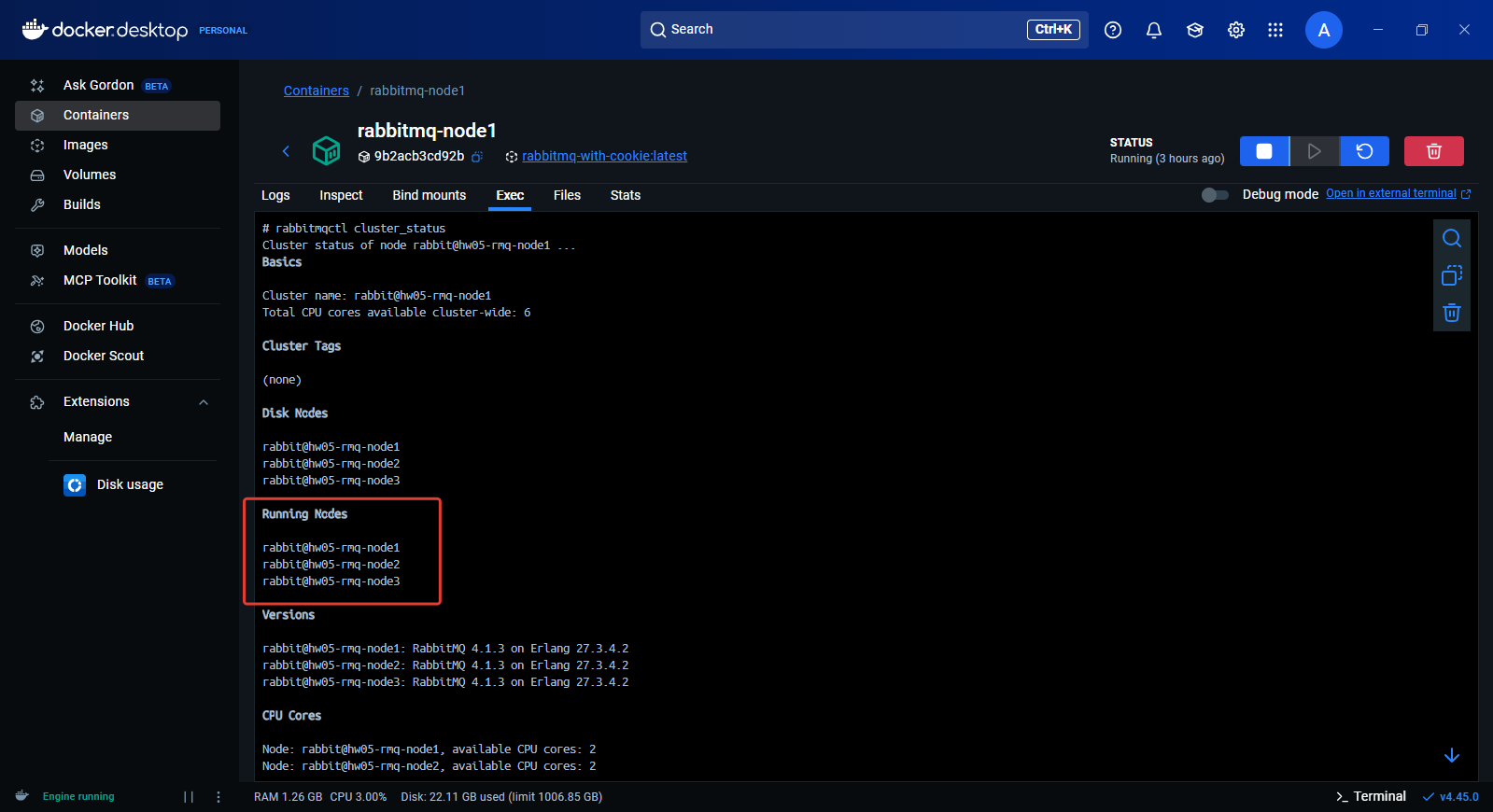
1. Добавляю ноды rabbitmq-node2 и rabbitmq-node3 последовательно в кластер, используя команду “**rabbitmqctl join\_cluster**”.



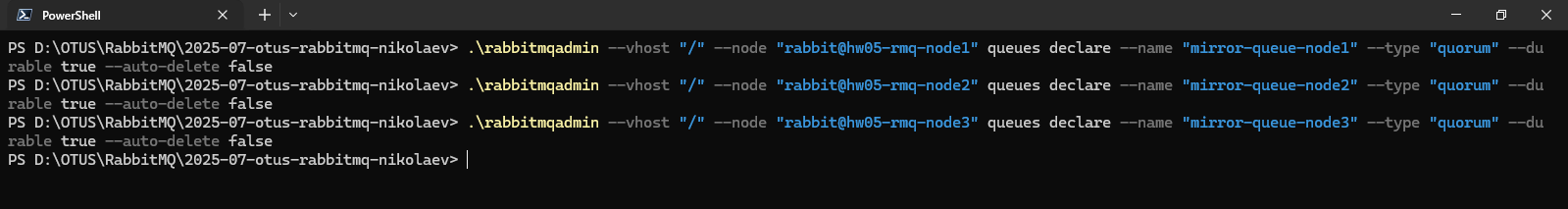
1. Убедился в том, что кластер собран и состоит из трех нод.



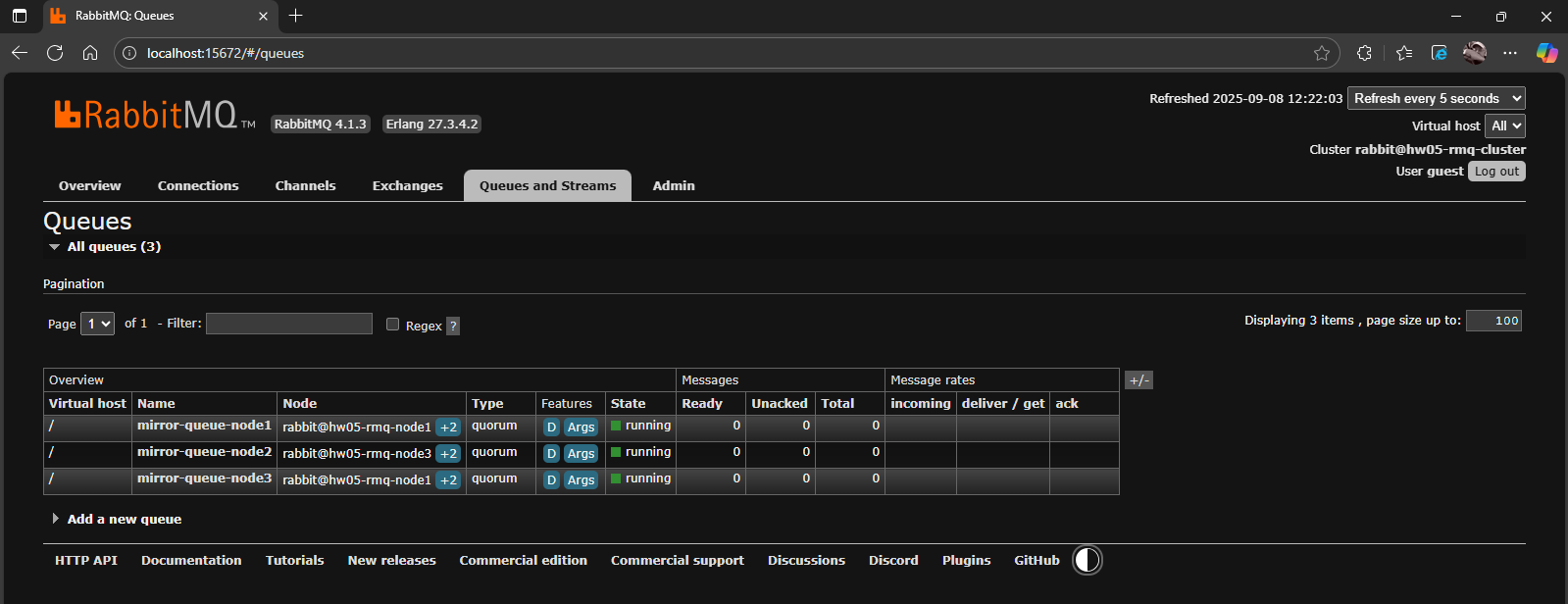
1. И через командную строку, “**rabbitmqctl cluster\_status**”.



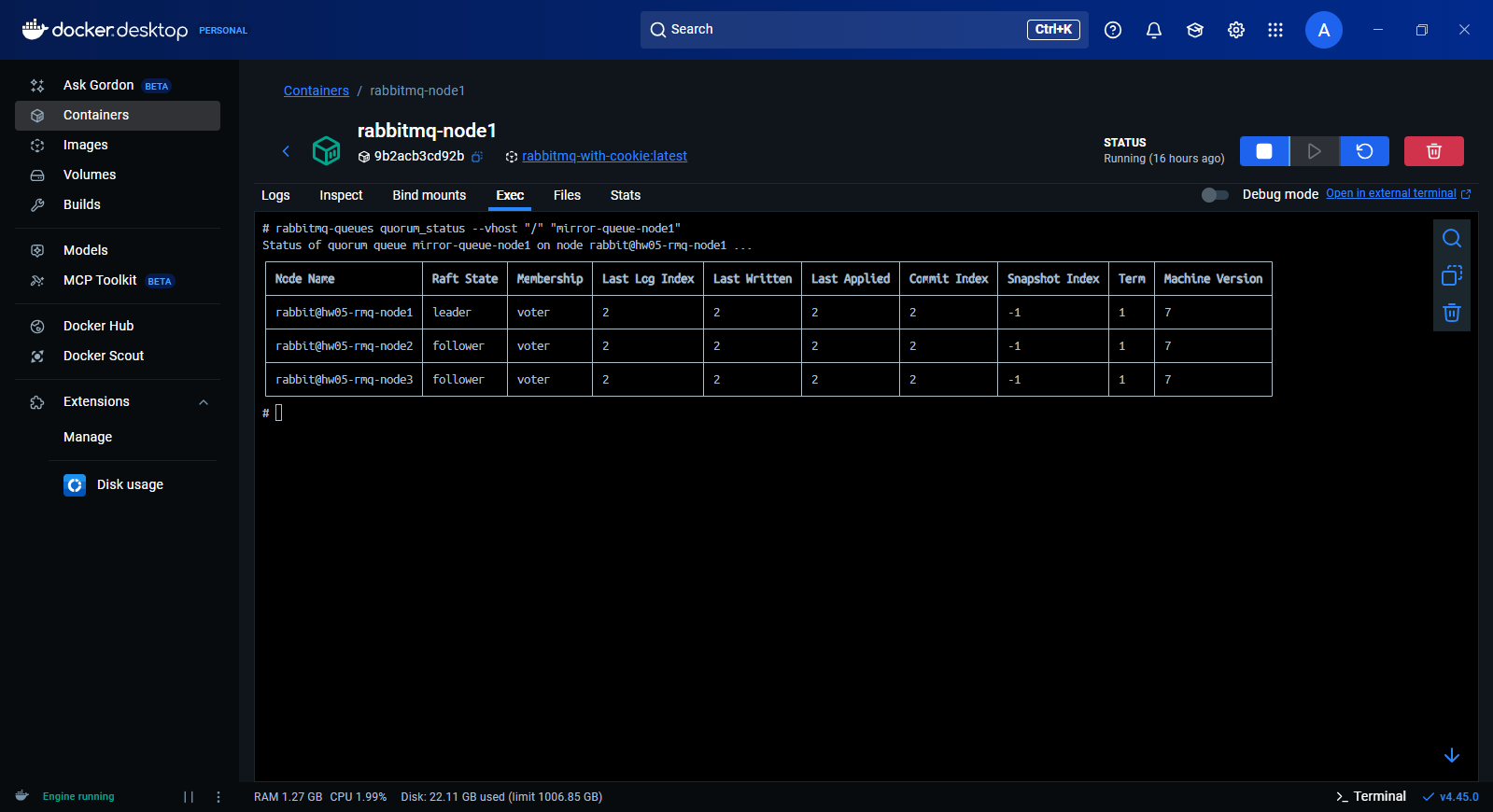
1. Создаем 3 очереди с типом “quorum” по количеству нод кластера.



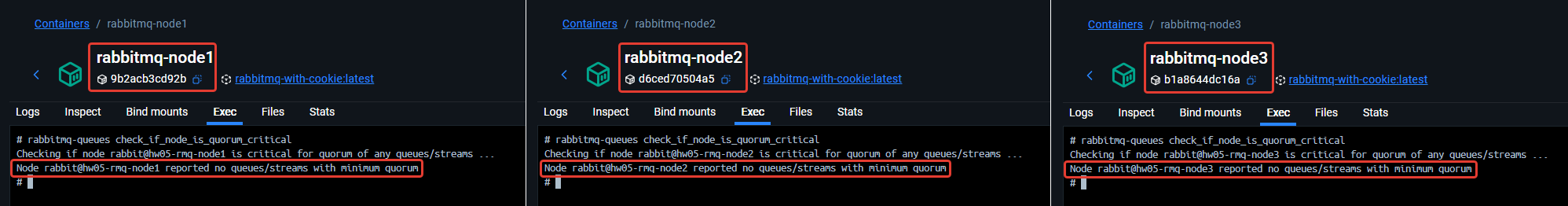
1. Проверяем что очереди создались



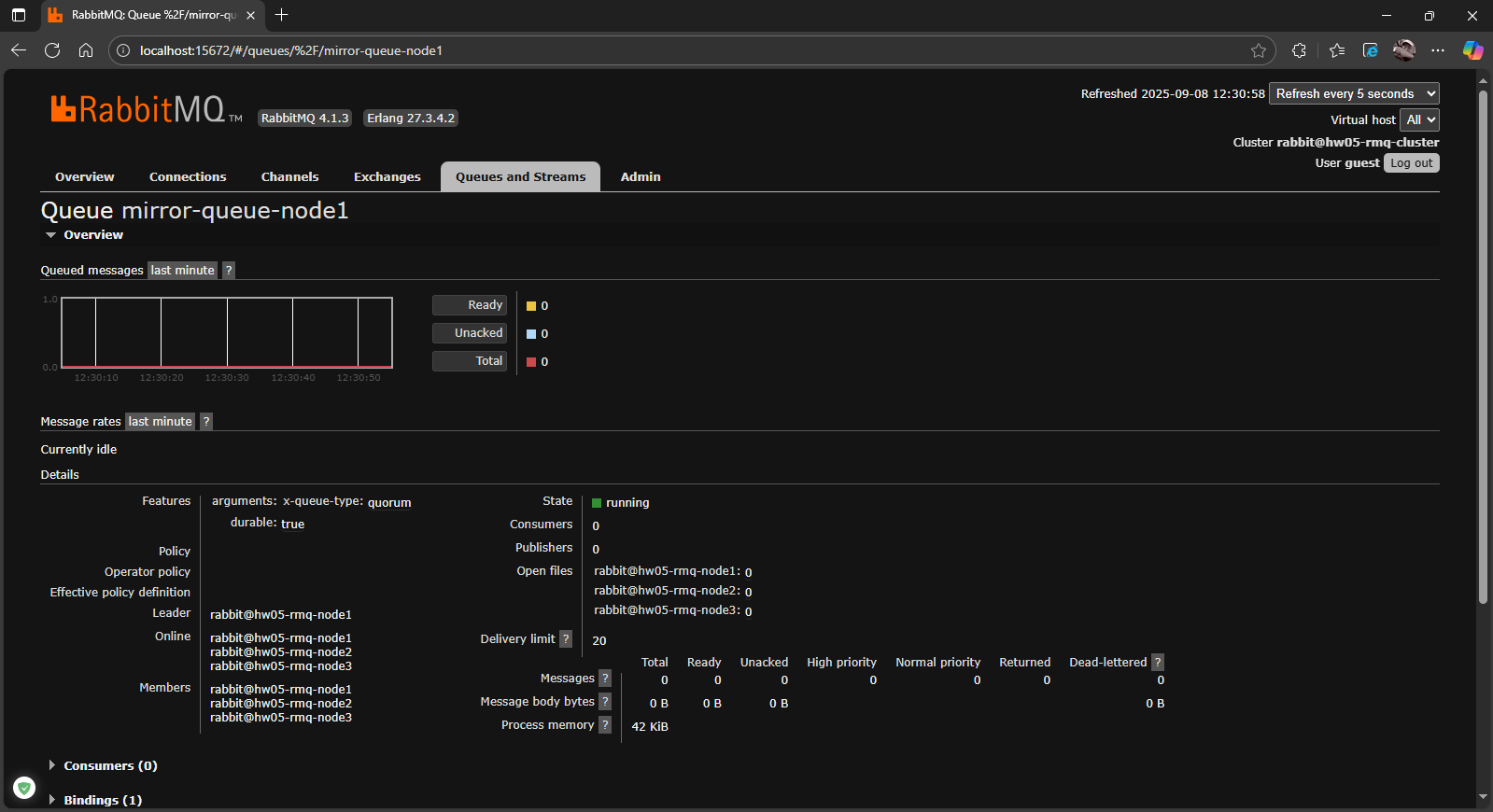
1. Проверяем статус кворума



1. Проверяем, являются ли ноды критичными для кворума любых очередей



1. Считаем, что все ноды и очереди созданы, находятся онлайн и реплицируются



## Выводы.

Т.к. я использовал последнюю версию RabbitMQ (4.1.3 на текущий момент), то создать классические зеркальные очереди не представляется возможным. Начиная с 4.0 версии RabbitMQ этот функционал полностью удален и взамен предлагается использовать кворумные очереди. В рамках выполнения домашнего задания я их и использовал.