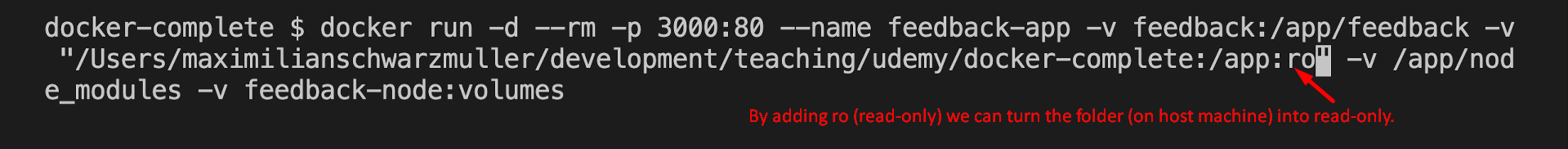
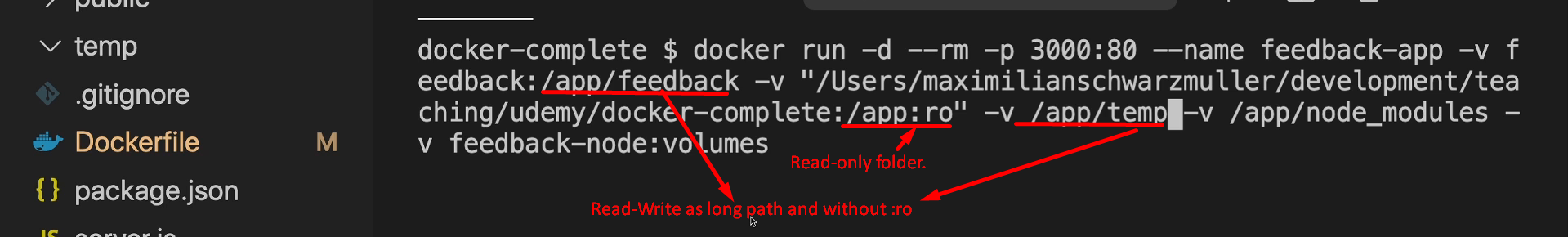
1. **Agenda**:
   1. Read-Only Volume
2. By default, Volumes are **read-write** so a container can write there.
3. In case of **Bind Mount**, the idea is that we should be able to change inside the container from outside (from the code base on host machine). But we know that volumes are by default Read-Write.
4.   
   “C:\jatin\Practice\Docker\data-volumes-03-adj-node-code\data-volumes-03-adj-node-code:/app**:ro**”   
   Basically, this **ro** (read-only) will make the /app folder inside the container read-only. So, the mapped path on Host Machine will also become read-only for the container.   
   Now docker/container will not be able to write into this folder (“C:\jatin\Practice\Docker\data-volumes-03-adj-node-code\data-volumes-03-adj-node-code”) or sub-folder.
5. But we can notice that /app/temp and /app/feedback folders must be writable as app needs to store files there.   
   **Solution**: Define volume for those paths as longer path wins in read-write concept too (as long path wins in bind mount).  
   NOTE: We have already -v feedback:/app/feedback so this is longer path than “C:\jatin\Practice\Docker\data-volumes-03-adj-node-code\data-volumes-03-adj-node-code:**/app:ro**” So, this longer path will win and it is writable as we didn’t apply **:ro**.  
   Define anonymous for /app/temp:  
   -v /app/temp  
   Why anonymous?  
   As we don’t want it to survive container restart.  
   
6. **NOTE**: During Bind Mount, adding :ro (read-only) will make the mapped folder inside the container read-only not the host machine folder, which is being mounted into the container but eventually, the host machine will become effectively read-only as container folder is read-only. Try to understand 😊  
   **NOTE**: If you add :ro during bind mount, you may get exception related to **node\_modules**.   
   **Solution**: Create empty node\_modules in codebase.