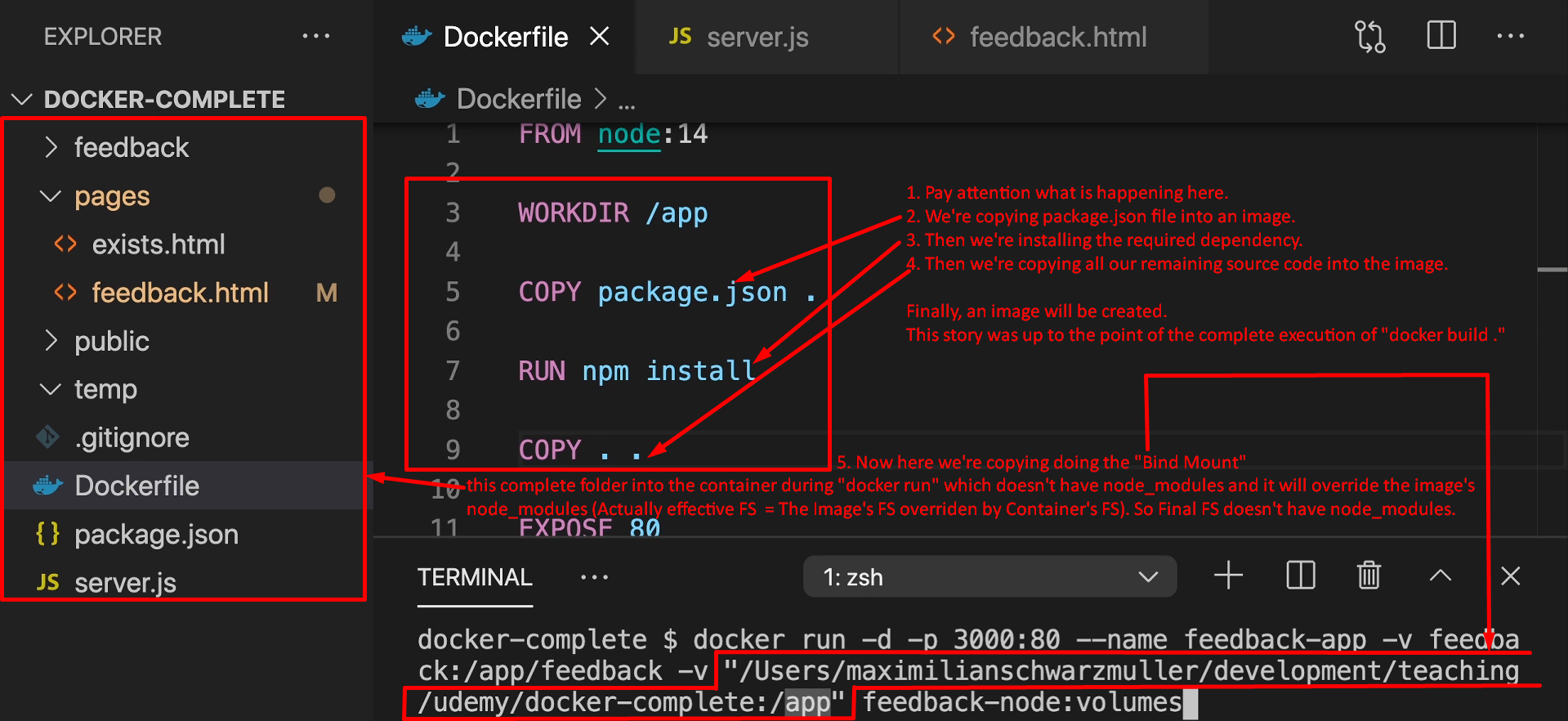
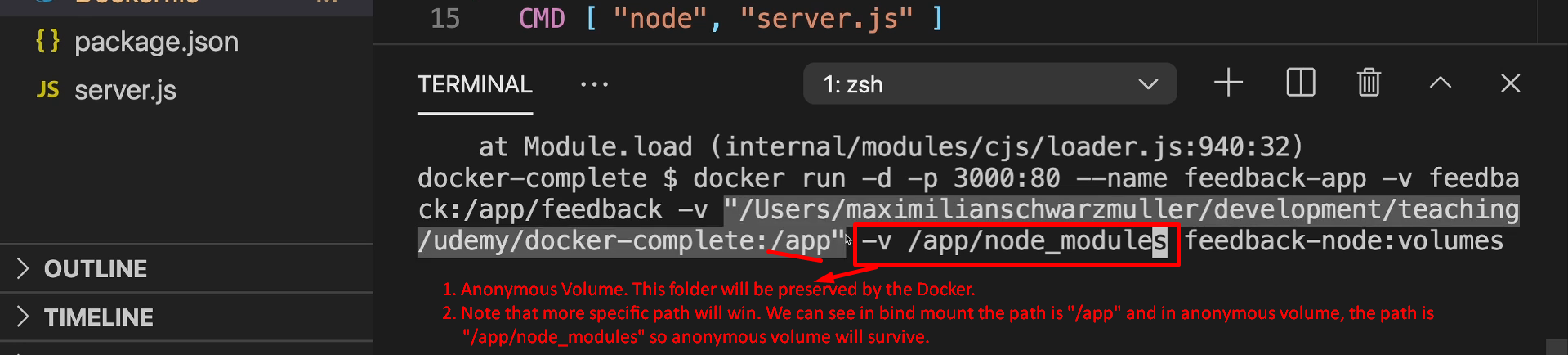
1. Let’s first understand how **containers** interact with **Volumes and Bind Mounts**.
2. Suppose we have both **Volume** and **Bind Mount** which we want to mount into our container.
3. We mount the “named volume” or “Bind Mount” with option -v with **“docker run”**.
4. 
5. **Solution**:
   1. Basically, we need to preserve node\_modules from being overridden.
   2. This can be achieved via **anonymous volume**.
   3. 
6. **Brief**: We want to make changes on our local system (host) and wanted those changes to happen inside the container rather than rebuilding the image with new changes. So, we approached **Bind Mount**. But we’re facing issue in **Bind Mount** like some parts inside container’s folder were removed. So to preserve those contents, we also applied anonymous volume.
7. **Why do we face overwrite issue in case of Bind Mount but not in case of Volume?**
8. Because in case of Volume the path on the host machine is managed by Docker itself & we don’t have any knowledge about the location and there will be nothing at that location. So, the file/folder movement direction is from container towards Host Machine’s Drive.   
   But in case of **Bind Mount**, as we specify the location on the host machine, and we have that location and whatever we put there is moved into container so we can override the content of the container. So, in case of **Bind Mount**, the files/folders direction is both sides. Actually, we’re commanding that copy everything from this folder (on host) into the container’s this folder and container’s folder’s previous content (everything) will be removed.