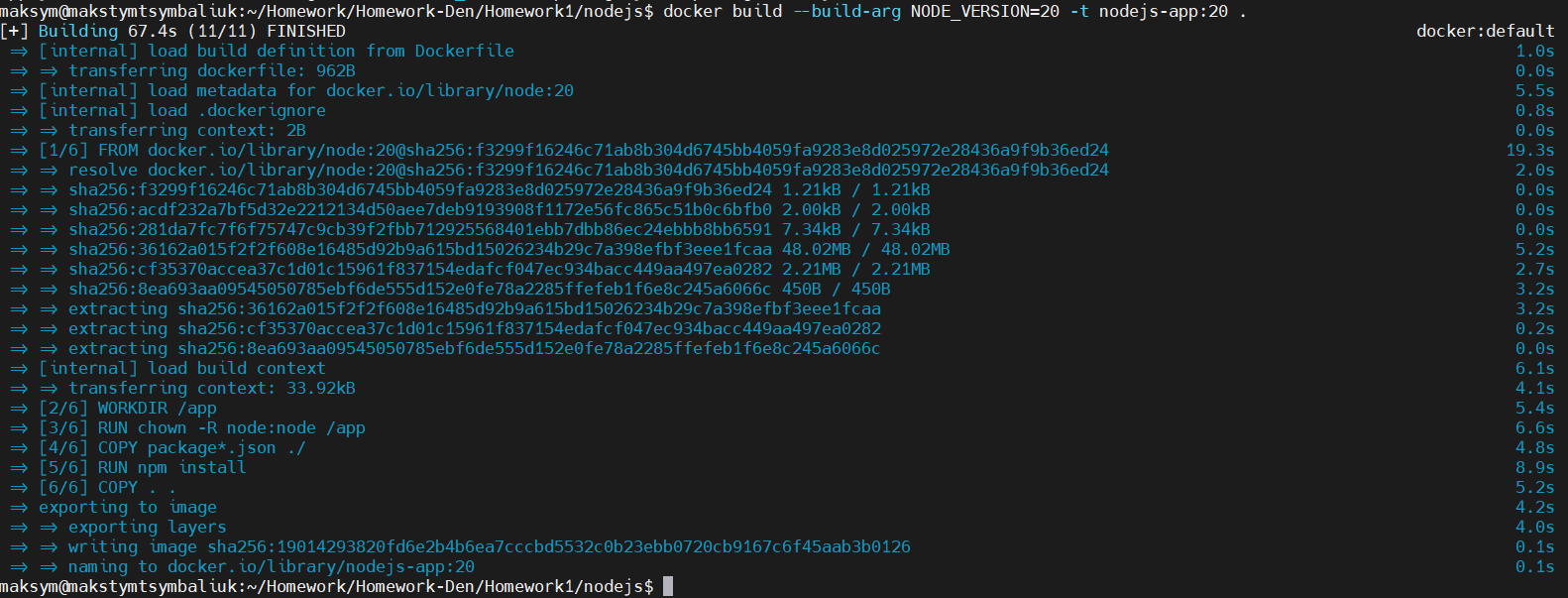
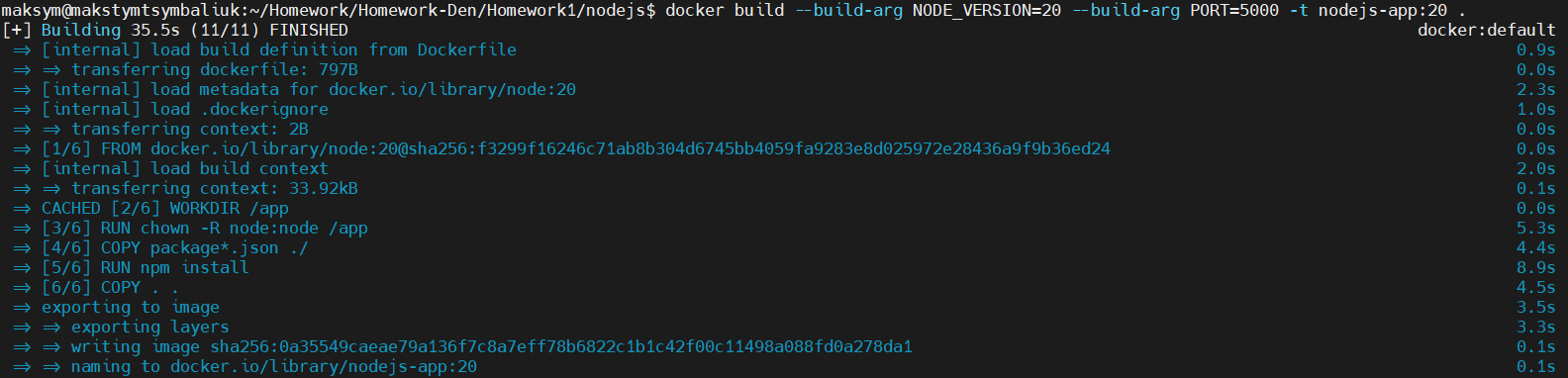
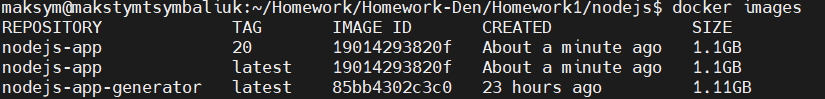
* Generate and add package-lock.json file to git



* Build container image with different base image tags using build args, e.g. node:20 and node:latest and mark resulting messages with same tags nodejs-app:20 and nodejs-app:latest





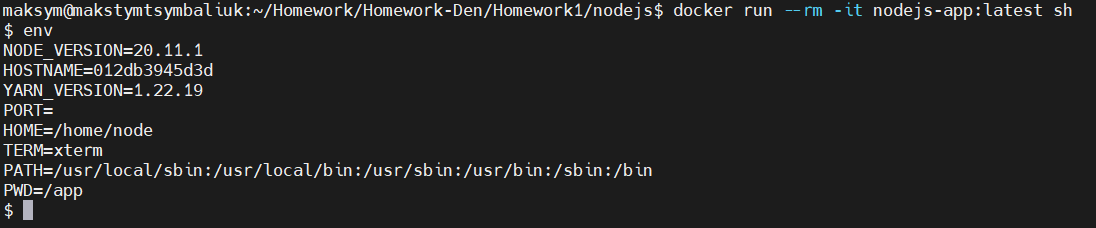


* Run nodejs-app:latest in interactive mode, check available env variables and exit, the container should be automatically removed from the docker ps -a list after that

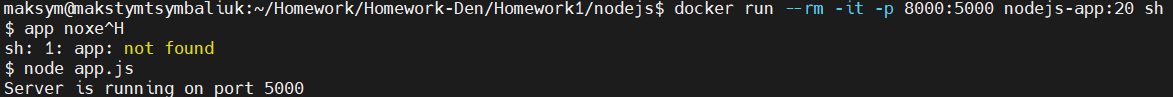


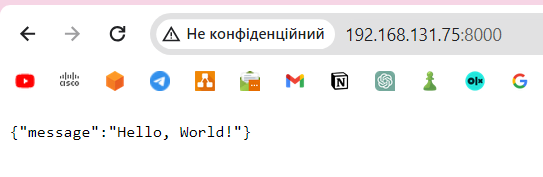


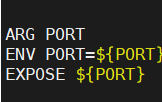
* Run the nodejs-app container interactively and control the port it is using with ENV variables, change it to 5000, as a plus, use port 8000 to access app in your browser



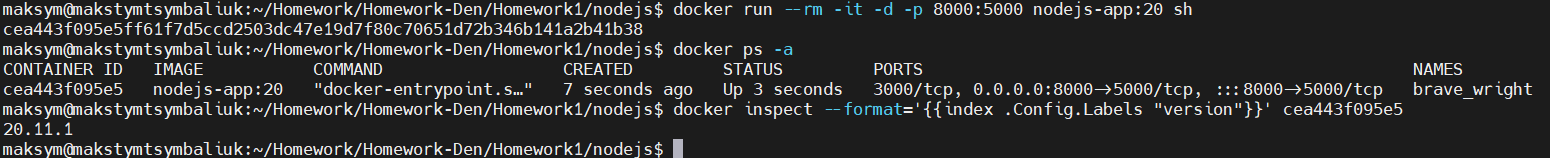
* Adjust Dockerfile to be able to control the used port during image build, the port defined on build should be used on run without additional configration
* Run the nodejs-app container interactively and control the port it is using with ENV variables, change it to 5000, as a plus, use port 8000 to access app in your browser



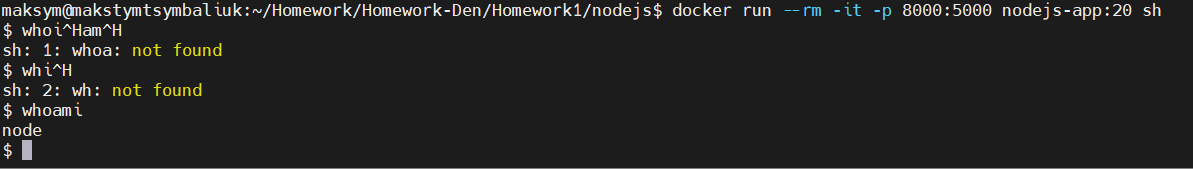




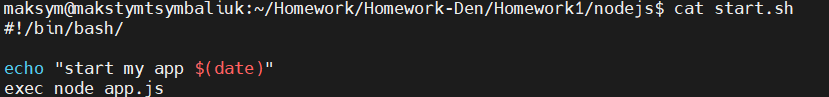
* Adjust Dockerfile to add labels to image, run it and use docker inspect to find them in container metadata



* Add creation of app non-root user and configure to use that user for the app



* Write a bash script which will print date to output before starting the app, run container and mount that file, specify it as entrypoint of the container, if it works fine, add as default entrypoint in Dockerfile



* Adjust Dockerfile and add health checks to make sure the app is alive, use the /healthz location for it
* Add /uuid location to the app which will return generated uuid values {"uuid": "XXXXX"}, install and use the [uuid](https://github.com/uuidjs/uuid" \t "_blank) module for that
* Create a new file Dockerfile.generator where install app using these [instructions](https://expressjs.com/en/starter/generator.html" \t "_blank), run it and confirm that app works in your browser, if possible use port 8080



