Steps to log into dependencyTrack:

1. #Command to pull dependency Track:
docker pull dependencytrack/bundled

2. # Creates a dedicated volume where data can be stored outside the container
docker volume create --name dependency-track

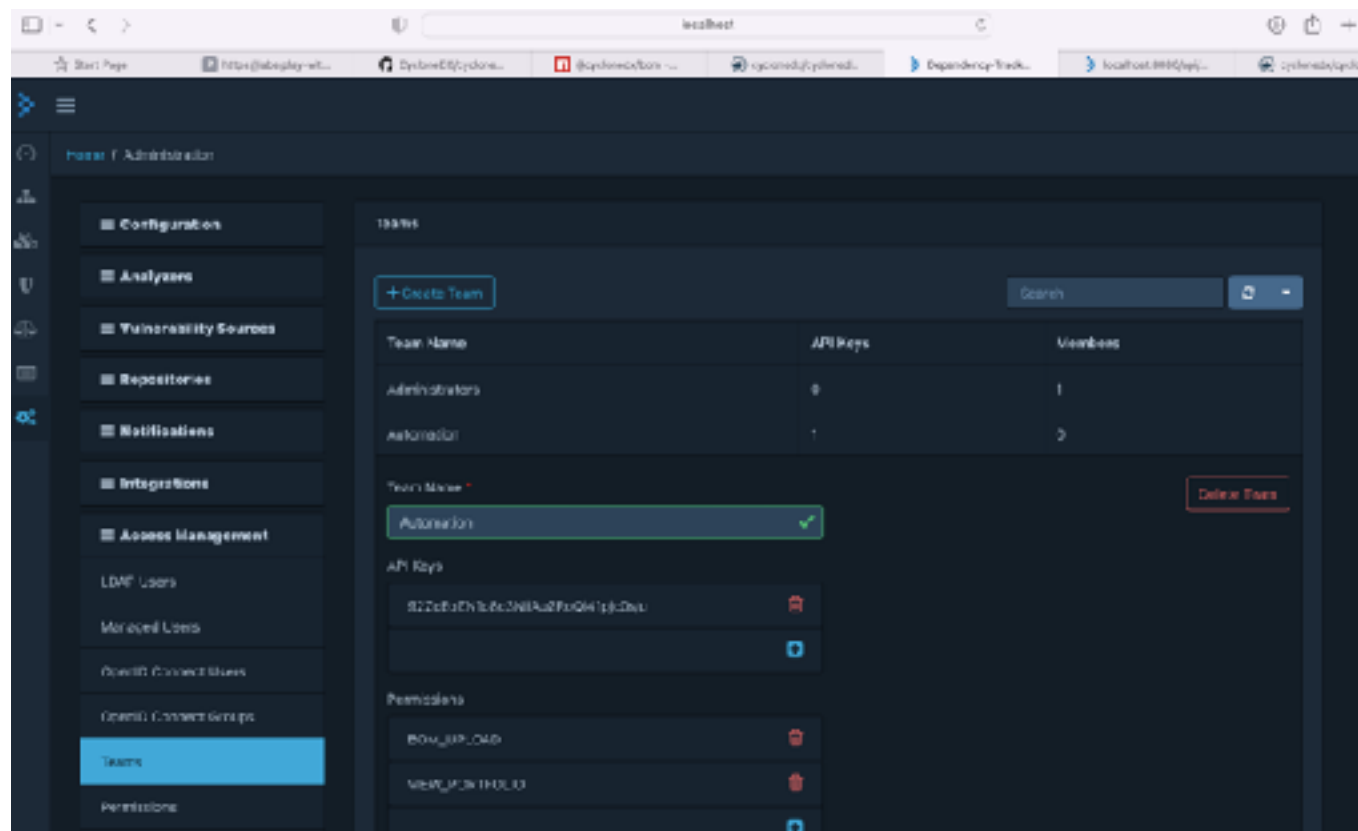3. # Run the bundled container with 8GB RAM on port 8080
docker run -d -m 8192m -p 8080:8080 --name dependency-track -v dependency-track:/data dependencytrack/bundled

4. **Launch following url in browser,** URL: http://localhost:8080/dashboard
5. After logging into password, we have to re-set the password.
6. Create a new project from 'Projects' section

**7. Make a note of 'API keys' and project id from URL after creating a project**

8.Command to get base64 encoded data of bom.xml:

$cat bom.xml | base64 -

PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiPz4KPGJvbSB4bWxucz0iaHR0cDovL2N5Y2xvbmVkeC5vcmcvc2NoZW1hL2JvbS
8xLjEiPgogICAgPGNvbXBvbmVudHM+CiAgICAgICAgPGNvbXBvbmVudCB0eXBlPSJsaWJyYXJ5Ij4KICAgICAgICAgICAgPGdyb3VwPmFmbTg5
PC9ncm91cD4KICAgICAgICAgICAgPG5hbWU+c3RhY2tY29yczwvbmFtZT4KICAgICAgICAgICAgPHZlcnNpb24+MS4zLjA8L3ZlcnNpb24+Ci
AgICAgICAgICAgIDxkZXNjcmlwdGlvbj5Dcm9zcy1vcmlnaW4gcmVzb3VyY2Ugc2hhcmluZyBsaWJyYXJ5IGFuZCBzdGFjayBtaWRkbGV3YXJl
PC9kZXNjcmlwdGlvbj4KICAgICAgICAgICAgPGxpY2Vuc2VzPgogICAgICAgICAgICAgICAgPGxpY2Vuc2U+CiAgICAgICAgICAgICAgICAgICAg
AgPGlkPk1JVDwvaWQ+CiAgICAgICAgICAgICAgICA8L2xpY2Vuc2U+CiAgICAgICAgICAgIDwvbGljZW5zZXM+CiAgICAgICAgICAgIDxwdXJs
PnBrZzpzc2pvb3Nlci9hc204OS9zdGFjay1jb3JzQDEuMy4wPC9wdXJsPgogICAgICAgIDwvY29tcG9uZW50PgogICAgICAgIDxjb21wb25lbn
QgdHlwZT0ibGlicmFyeSI+CiAgICAgICAgICAgIDxncm91cD5icmljazwvZ3JvdXA+CiAgICAgICAgICAgIDxuYW1lPmhhdGg8L25hbWU+CiAg
ICAgICAgICAgIDx2ZXJzaW9uPjAuOS4yPC92ZXJzaW9uPgogICAgICAgICAgICA8ZGVzY3JpcHRpb24+QXJiaXRyYXJ5LXByZWNpc2lvbiBhcm
l0aG1ldGljIGxpYnJhcnk8L2Rlc2NyaXB0aW9uPgogICAgICAgICAgICA8bGljZW5zZXM+CiAgICAgICAgICAgICAgICA8bGljZW5zZT4KICAg
ICAgICAgICAgICAgICAgICA8aWQ+TUlUPC9pZD4KICAgICAgICAgICAgICAgIDwvbGljZW5zZT4KICAgICAgICAgICAgPC9saWNlbnNlcz4KIC
AgICAgICAgICAgPHB1cmw+cGtnOmNvbXBvc2VyL2JyaWNrL21hdGhAMC45LjI8L3B1cmw+CiAgICAgICAgPC9jb21wb25lbnQ+CiAgICAgICAg
PGNvbXBvbmVudCB0eXBlPSJsaWJyYXJ5Ij4KICAgICAgICAgICAgPGdyb3VwPnRvY3RyaW5lPC9ncm91cD4KICAgICAgICAgICAgPG5hbWU+aW
5mbGVjdG9yPC9uYW1lPgogICAgICAgICAgICA8dmVyc2lvbj4yLjAuMzwvdmVyc2lvbj4KICAgICAgICAgICAgPGRlc2NyaXB0aW9uPlBIUCBE
b2N0cmluZSBJbmZsZWN0b3IgaXMgYSBzbWFsbCBsaWJyYXJ5IHRoYXQgY2FuIHBlcmZvcm0gc3RyaW5nIG1hbmlwdWxhdGlvbnMgd2l0aCByZW
dhcmQgdG8gdXBwZXIvbG93ZXJjYXNlIGFuZCBzaW5ndWxhci9wbHVyYWwgZm9ybXMgb2Ygd29yZHMuPC9kZXNjcmlwdGlvbj4KICAgICAgICAg
ICAgPGxpY2Vuc2VzPgogICAgICAgICAgICAgICAgPGxpY2Vuc2U+CiAgICAgICAgICAgICAgICAgICAgPGlkPk1JVDwvaWQ+CiAgICAgICAgIC
AgICAgICA8L2xpY2Vuc2U+CiAgICAgICAgICAgIDwvbGljZW5zZXM+CiAgICAgICAgICAgIDxwdXJsPnBrZzpjb21wb3Nlci9kb2N0cmluZS9p
bmZsZWN0b3JAMi4wLjM8L3B1cmw+CiAgICAgICAgPC9jb21wb25lbnQ+CiAgICAgICAgPGNvbXBvbmVudCB0eXBlPSJsaWJyYXJ5Ij4KIC………
……………………..etc

9. Command to upload BOM as base64 encoded data: curl -v -X "PUT" "http://localhost:8080/api/v1/bom" -H 'Content-Type: application/json' -H 'X-API-KEY: 92ZcEoZN1o8c3NiiAuZRoQi41pjc0sju' -d $'{"project": "758fe80a-ae76-451b-a298-ac2816b5c5f5", "bom": "<Copy/Paste above response here>"}'

10. After executing above command successfully, Re-fresh browser and observe below response.
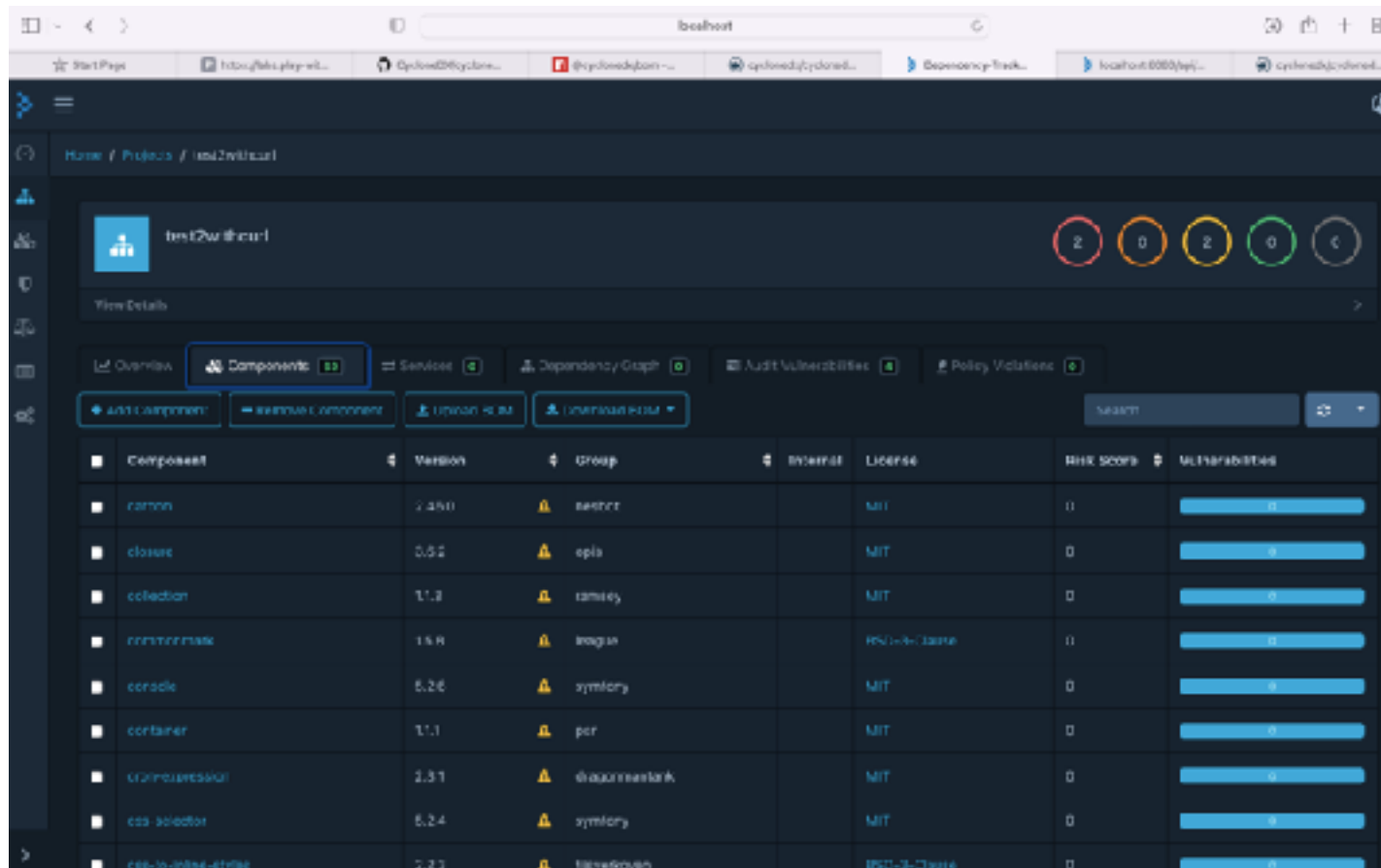
Dockerfile:

```
From node:12

RUN apt-get update

RUN apt install -y git

RUN apt install -y curl

#RUN apt install -y npm
```

```
RUN git clone https://github.com/CycloneDX/cyclonedx-node-module.git

RUN git clone https://github.com/CycloneDX/cyclonedx-core-java.git

#WORKDIR /cyclonedx-node-module
# get install script and pass it to execute:
#RUN curl -sL https://deb.nodesource.com/setup_4.x | bash
# and install node
#RUN apt-get install nodejs
# confirm that it was successful
#RUN node -v
# npm installs automatically
RUN npm -v

RUN npm install

RUN npm install -g @cyclonedx/bom

RUN curl -sSfL https://raw.githubusercontent.com/anchore/syft/main/install.sh | sh -s -- -b /usr/local/bin
RUN curl -sSfL https://raw.githubusercontent.com/anchore/grype/main/install.sh | sh -s -- -b /usr/local/bin
```

docker-compose.yml:
```
version: "3.9"
services:
        web:
                build: .
        cyclonedx:
                image: "cyclonedx/cyclonedx-cli"
                #       dotnet:
                #image: "cyclonedx/cyclonedx-dotnet"
                #python:
#       cyclonepython:
#               image: "cyclonedx/cyclonedx-python"
              # apt: python-pip3
#               pip: freeze > requirements.txt
```

```
                        # app:
                        #build: .
                        #dockerfile:  Dockerfile
                        #depends_on:
                        #          - cyclonedx
                        #cyclonedx-python:
            cyclonedx-node-module:
#                command: npm start
                        build: cyclonedx-node-module
#                 command: npm install

                        #cyclonedx-dotnet:
                        #build: cyclonedx-dotnet
            cyclonedot:
                        image: "cyclonedx/cyclonedx-dotnet"
#          cyclonedx-core-java:
#                build: cyclonedx-core-java
```

Creating Volume: $ sudo docker volume create vol2

Running container with volume: $docker run --rm -it --name test3 -v "$(pwd)"/WebGoat:/vol2 <Imagename>

Inspecting Volume to know container path: sudo docker volume inspect vol2

```
docker volume inspect vol2
[
    {
        "CreatedAt": "2022-02-16T16:10:57Z",
        "Driver": "local",
        "Labels": {},
        "Mountpoint": "/var/lib/docker/volumes/vol2/_data",
        "Name": "vol2",
```

```
        "Options": {},
        "Scope": "local"
    }
]
```

Download vulnerable web goat from GitHub and save it in local directory:

Git clone https://github.com/WebGoat/WebGoat.git

Command to create bom file with syft :

```
 docker run --rm -it --name test3 -v "$(pwd)"/WebGoat:/vol2 8a37256c9914 syft packages dir:/vol2  -o cyclonedx
```

```
Observe that bom file gets created.
```