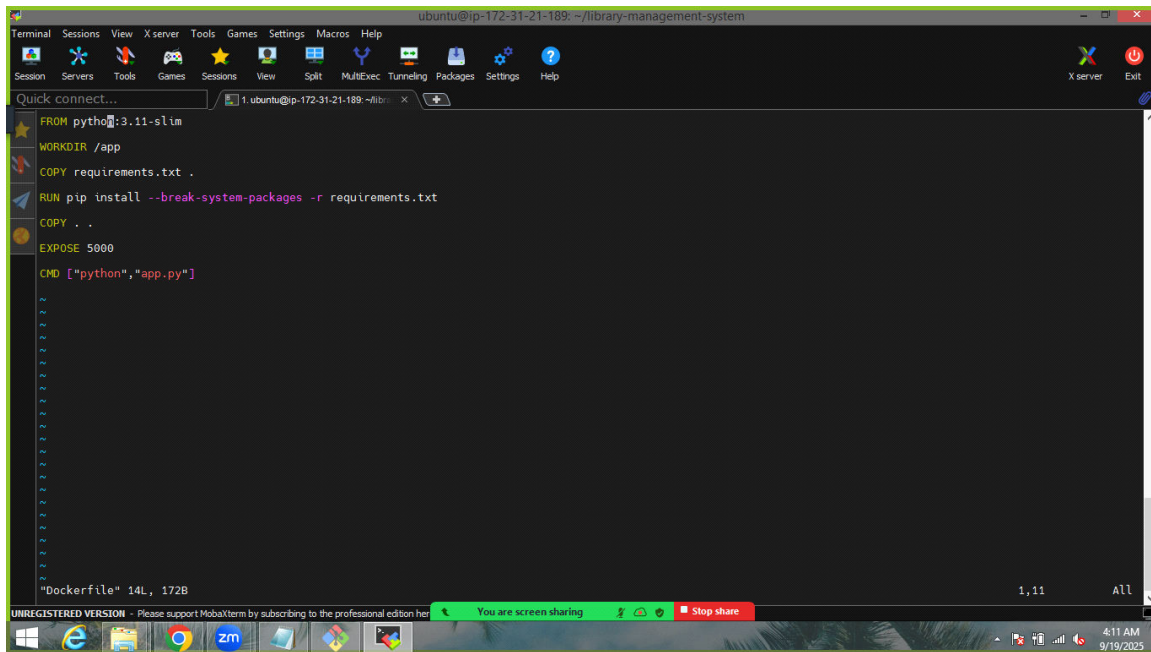


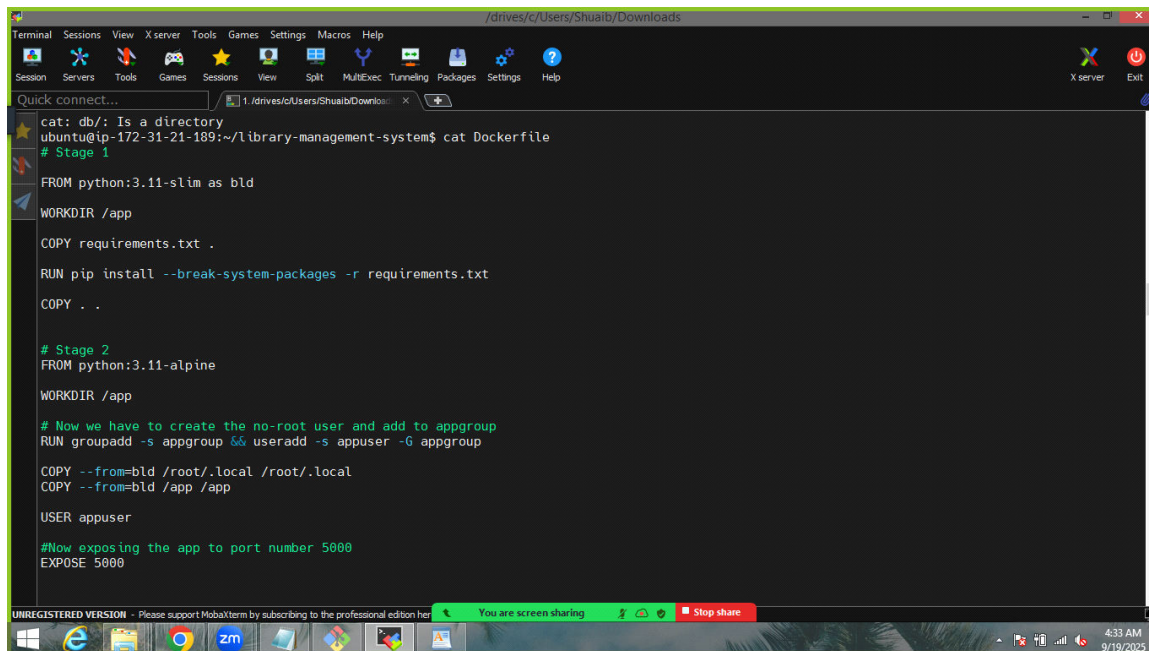
1: Wrote a Dockerfile for a Python flask application and exposed the container to port number 5000.



```
FROM python:3.11-slim
WORKDIR /app
COPY requirements.txt .
RUN pip install --break-system-packages -r requirements.txt
COPY . .
EXPOSE 5000
CMD ["python","app.py"]
```

The screenshot shows a terminal window titled 'ubuntu@ip-172-31-21-189: ~/library-management-system'. The terminal displays the contents of a Dockerfile, which includes instructions to use a Python 3.11-slim base image, set the working directory to /app, copy requirements.txt, install dependencies with pip, copy the application files, expose port 5000, and run the application using python app.py. The terminal also shows a 'Quick connect...' bar and a 'You are screen sharing' notification.

2: Multistage docker builds for the python library management flask app.



```
cat: db/: Is a directory
ubuntu@ip-172-31-21-189:~/library-management-system$ cat Dockerfile
# Stage 1
FROM python:3.11-slim as bld
WORKDIR /app
COPY requirements.txt .
RUN pip install --break-system-packages -r requirements.txt
COPY . .

# Stage 2
FROM python:3.11-alpine
WORKDIR /app

# Now we have to create the no-root user and add to appgroup
RUN groupadd -s appgroup && useradd -s appuser -G appgroup

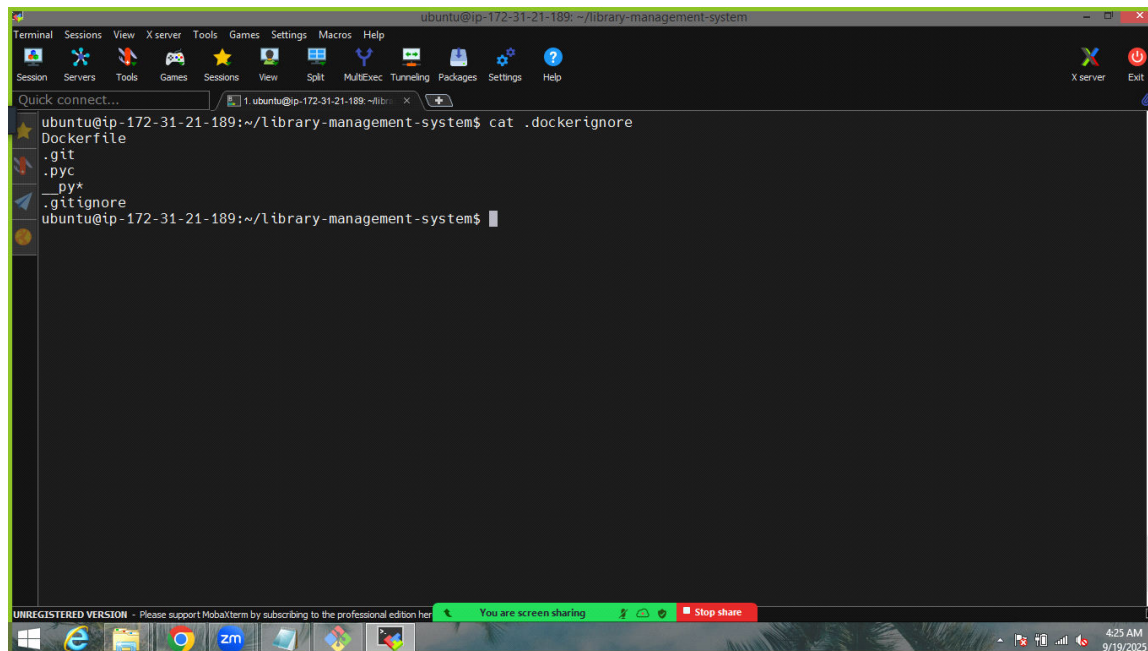
COPY --from=bld /root/.local /root/.local
COPY --from=bld /app /app

USER appuser

#Now exposing the app to port number 5000
EXPOSE 5000
```

The screenshot shows a terminal window titled '/drives/c/Users/Shuaib/Downloads'. The terminal displays the contents of a multistage Dockerfile. It starts with a message 'cat: db/: Is a directory' and then shows the command 'ubuntu@ip-172-31-21-189:~/library-management-system\$ cat Dockerfile'. The Dockerfile content includes two stages: Stage 1 (building the application) and Stage 2 (creating a non-root user and exposing the application). The terminal also shows a 'Quick connect...' bar and a 'You are screen sharing' notification.

Also included the .dockerignore file to exclude the spare files which are not necessary for application.

A screenshot of a MobaXterm terminal window. The title bar reads 'ubuntu@ip-172-31-21-189: ~/library-management-system'. The terminal shows the command 'cat .dockerignore' being executed, with the output listing files and patterns to be ignored: 'Dockerfile', '.git', '.pyc', '.py\*', and '.gitignore'. The prompt 'ubuntu@ip-172-31-21-189:~/library-management-system\$' is visible at the bottom of the terminal. The MobaXterm interface includes a menu bar (Terminal, Sessions, View, X server, Tools, Games, Settings, Macros, Help), a toolbar with icons for Session, Servers, Tools, Games, Sessions, View, Split, MultiExec, Tunneling, Packages, Settings, and Help, and a 'Quick connect...' search bar. A status bar at the bottom indicates 'UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here', 'You are screen sharing', and a 'Stop share' button. The system tray shows the date and time as '4:25 AM 9/19/2025'.

### 3: Differences between image vs container vs volume vs network in Docker.

**image:** A docker image is the blueprint or template from we create the containers.

**container:** A container is the running instance of an image that includes all the dependences of the application.

**volumes:** Volumes are like attached storage device to the container like if the container gets crashed the data does not lose i.e. volumes provide data persistent.

**network:** Docker networking is the concept how containers communicate with each other.