How to **Dockerize** a Flask Python application

ClickIT
DevOps & Software Development

Setting up a machine manually to deploy your Python Flask Application multiple times can easily lead to human errors and also increases the chances of missing certain dependencies. It takes plenty of time to figure out the errors, fix them, and then deploy the applications.

ClickIT
DevOps & Software Development

Deploy your Flask python application using Docker in Production.

01 ▸ First and foremost, never build your Docker Images on the Production servers. You must always pull your images from the central Docker Registry/Repository.

02 ▸ Always verify the source of base images that you use in your Dockerfiles.

03 ▸ Clean up containers that are no longer running using the “docker rm” command.

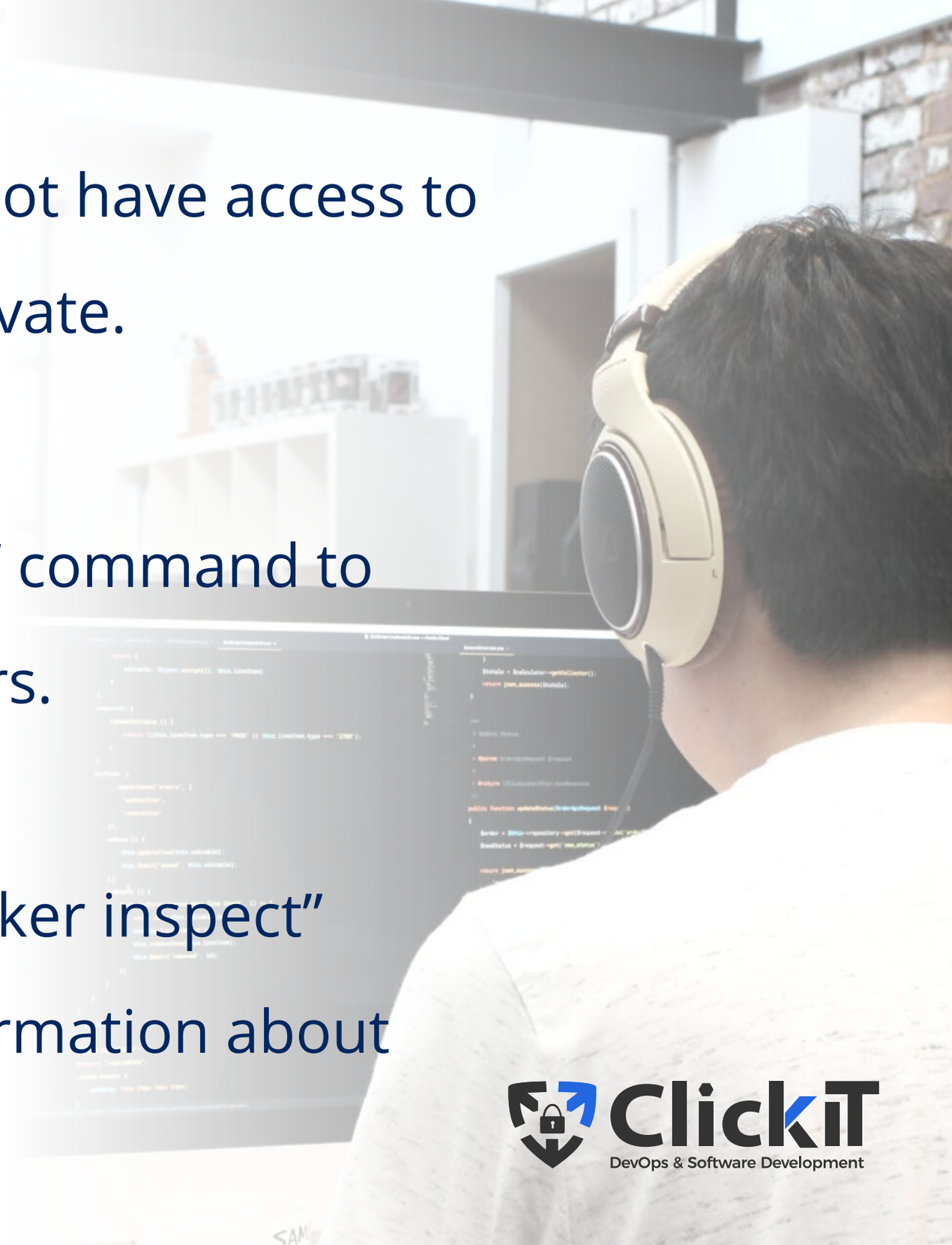
04 ▸ Use volumes to store your application logs on persistent volumes.

ClickIT
DevOps & Software Development

05 ▸ Public traffic should not have access to certain containers that are private.

06 ▸ Use the “docker logs” command to fetch logs from your containers.

07 ▸ Make use of the “docker inspect” command to get detailed information about Docker Objects.



ClickIT
DevOps & Software Development

08 ▸ Use “docker secret” to manage credentials or secrets, avoid storing secrets in Plain Text format.

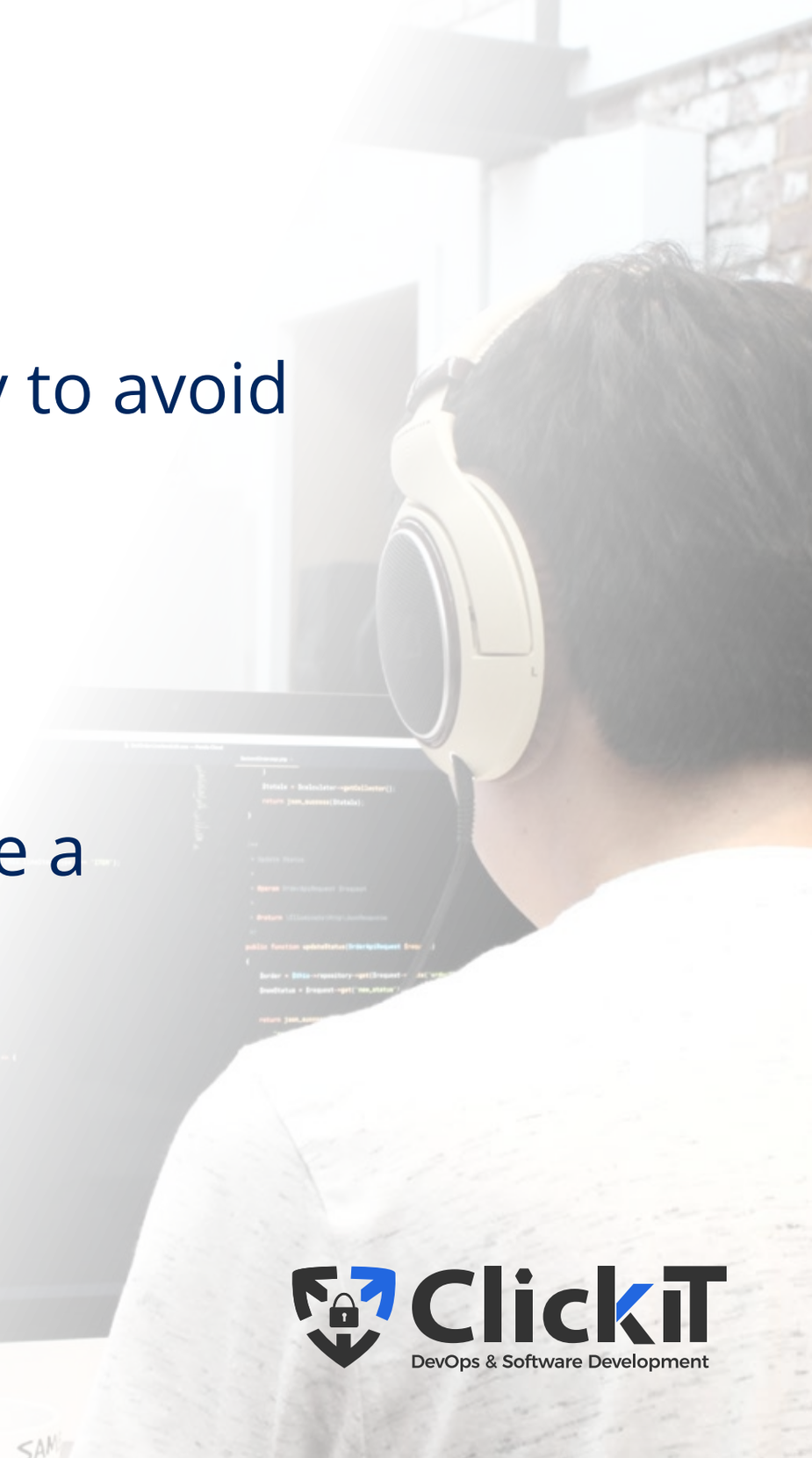
09 ▸ Limit resources for containers, enforce resource limits so that containers use no more than a given limit.

10 ▸ Enable logging and monitoring for your containerized applications.

ClickIT
DevOps & Software Development

11 ▸ Use the “restart: always” policy to avoid downtime.

12 ▸ And very important to note, use a Container Orchestration Tool.



Check out this link for more updates

https://www.clickittech.com/devops/dockerize-flask-python-application/

ClickIT
DevOps & Software Development