**Report on UnifyAI Training Concepts:**

**Python:  
Concept Summary:**  
Python is the foundation of my AI and DevOps journey. It’s used for backend development, automation, and AI/ML scripting.

**What I’ve Done:**

* Practiced key Python concepts such as loops, conditionals, functions, and file handling.
* Solved DSA problems like Kth smallest/largest element, binary search in rotated arrays, and string mismatch problems.
* Worked with Python libraries like **NumPy, Pandas, Matplotlib, Seaborn, Scikit-learn**

**What I’ve Understood:**

* I’ve developed a strong grasp of Python syntax and problem-solving.
* I understand libraries used for **data analytics**, **visualization**, and **machine learning**.
* I can integrate Python scripts with **Flask**, **Streamlit**, and AI models.

**Git, GitHub, VCS:**

**Concept Summary:**  
Git is a distributed version control system, and GitHub is the remote platform for collaboration and repository management.

**What I’ve Done:**

* Learned essential Git commands: git init, clone, add, commit, push, pull, branch, merge, and rebase.
* Understood the difference between **Git Rebase** and **Merge**.
* Pushed Docker and project repositories to GitHub successfully.
* Configured **Git login in CMD** and managed SSH authentication for GitHub.

**What I’ve Understood:**

* I clearly understand how version control tracks project changes.
* I’m comfortable with Git branching, commits, and resolving conflicts.
* I can manage repositories and collaborate on GitHub effectively.

**Docker:  
Concept Summary:**  
Docker is a platform that packages applications into lightweight, portable containers.

**What I’ve Done:**

* Pulled, built, tagged, and pushed Docker images to Docker Hub.
* Used commands like docker run, ps, logs, exec, commit, build, compose, and network.
* Understood the concepts of docker tag and docker compose for service orchestration.
* Learned about optimizing image size using **multi-stage builds** and **slim base images**.
* Deployed Flask apps inside containers using port mapping (-p 5000:5000) and volumes.

**What I’ve Understood:**

* I have a strong understanding of the **container lifecycle**, **volumes**, and **Docker Compose**.
* I can troubleshoot container issues using logs and inspect commands.
* I know how to build efficient and portable Docker images for deployment.

**Linux:  
Concept Summary:**  
Linux is the backbone of servers, containers, and cloud infrastructure.

**What I’ve Done:**

* Practiced essential Linux commands (ls, cd, du -sh, df -h, ps aux, lsof, kill, etc.).
* Created and managed users, groups, and permissions.
* Worked on **cron jobs** (creating, listing, and stopping them).
* Explored **firewall**, **networking**, and **SELinux** concepts.
* Managed **disk partitions**, **mounting**, and **system services** using systemctl.

**What I’ve Understood:**

* I’m comfortable navigating and administering Linux systems.
* I know how to manage processes, storage, and user permissions.
* I can handle network and firewall configurations for applications.

**Containerization:**

**Concept Summary:**  
Containerization packages applications and their dependencies into isolated environments for consistent deployment across systems. Tools like **Docker** and **Kubernetes (K8s)** make it easier to manage, scale, and orchestrate these containers efficiently.

**What I’ve Done:**

* Learned how **Docker containers** isolate applications from the host OS.
* Explored **container networking**, linking multiple services using **Docker Compose**.
* Compared **VMs vs Containers** in terms of performance, efficiency, and portability.
* Worked with **Kubernetes (K8s)** and **Minikube** for container orchestration.
* Installed and configured **kubeadm** to set up a **Kubernetes cluster** with **1 master node** and **2 worker nodes** across 3 Fedora VMs.
* Practiced essential Kubernetes commands such as:

***# Basic cluster and node management***

kubectl get nodes

kubectl get pods -A

kubectl describe node <node-name>

kubectl get deployments

***# Creating and managing deployments***

kubectl create deployment myapp --image=nginx

kubectl get all

kubectl expose deployment myapp --type=NodePort --port=80

***# Checking services and logs***

kubectl get svc

kubectl logs <pod-name>

kubectl exec -it <pod-name> -- /bin/bash

***# Cluster initialization with kubeadm***

sudo kubeadm init --pod-network-cidr=10.244.0.0/16

kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml

***# Joining worker nodes***

kubeadm join <master-ip>:6443 --token <token> --discovery-token-ca-cert-hash sha256:<hash>

Practiced **Minikube** for single-node Kubernetes environments using commands like:

minikube start

minikube status

minikube dashboard

minikube service myapp

minikube stop

minikube delete

**What I’ve Understood:**

* I understand how **container orchestration** automates scaling, load balancing, and deployment management.
* I can deploy and manage containers across multiple nodes using **Kubernetes**.
* I’ve gained hands-on experience with **Minikube** for local testing and **kubeadm** for multi-node cluster setup.
* I understand how to manage **pods, deployments, services**, and **networking** inside Kubernetes.

**Webservers:**

**Concept Summary:**  
Web servers handle HTTP requests and serve web applications.

**What I’ve Done:**

* Hosted Flask and Streamlit applications locally and within Docker containers.
* Worked with **Nginx** and **Apache** web servers for deploying and reverse proxying web applications.
* Learned about **ports (HTTP – 80, HTTPS – 443, SSH – 22)** and configured appropriate firewall rules.
* Understood backend app serving via flask run and containerized hosting.
* Configured port mapping and verified service accessibility through firewall rules and server configurations.

**What I’ve Understood:**

* I clearly understand how webservers manage client–server communication.
* I can configure and deploy applications using **Nginx** and **Apache**.
* I understand the importance of reverse proxying, load balancing, and open firewall ports for web accessibility.

**Cloud Fundamentals:**

**Concept Summary:**  
Cloud computing delivers on-demand computing services over the internet.

**What I’ve Done:**

* Studied **Cloud Models (IaaS, PaaS, SaaS)** and **Virtualization** concepts.
* Understood **VM instances**, **Public/Private IPs**, and **firewall port rules**.
* Explored **storage services** (S3, Blob, Buckets), **snapshots**, and **cost calculators**.
* Practiced **VM provisioning**, **firewall access**, and **cloud disk management**.
* Learned about **Ingress/Egress**, **security rules**, and **IAM (Identity & Access Management)**.

**What I’ve Understood:**

* I have a clear understanding of cloud concepts like scalability, cost efficiency, and resource allocation.
* I know how to deploy and secure applications on cloud VMs.
* I’m comfortable with configuring networking and storage in cloud environments.