Cloud Management System – Final Report

Team Members: Malak Mohamed, Hossam Abdelreheem, Abdelrahman Elshazly, Tarek Amr, Farah Walid, Linah Mohamed

Submission Date: 18/05/2025

Project Overview

The Cloud Management System provides a comprehensive platform for managing virtual machines and Docker containers through a graphical user interface. Building upon what we left off in Phase 1, which covered virtual disks and VM creation, the final version expands to support full Docker container management, including image building, container control, and Docker Hub integration.

This project was developed using Python and PySide6, and leverages QEMU and Docker to simulate real-world virtualization and cloud deployment environments in a user-friendly manner.

Implemented Features

a. Virtual Disk Management

- Supports dynamic or fixed disk allocation.
- Supports formats: qcow2, raw, img, vmdk, vdi, vhd, vhdx, etc.
- Allows disk creation with size/unit (e.g., 10G, 500M).
- Resize functionality for disk expansion (shrinking is blocked).
- File path validation and error handling.

b. Virtual Machine Creation

- Customizable options: VM name, CPU (1–4 cores), memory (512 MB–32 GB), disk, ISO.
- Validations: VM name rules, memory range, disk/ISO required.
- Launches VM using QEMU subprocess with GUI-based file selection.
- Supports multiple disk types and OS installations.

c. Docker File Creation

- Users can write Dockerfile content directly in a text box.
- GUI lets user save the Dockerfile to any desired path.
- Logs and confirmations shown in a text area.

d. Docker Image Build

- Build image from Docker file with user-provided image name.
- Uses docker build command internally.
- Output displayed in log area with error handling if Docker is not installed or fails.

e. List Docker Images

- Shows all Docker images installed on the system using docker images.
- Output formatted and displayed in the log area.

f. List Running Containers

- Shows currently active containers via docker ps.
- Full container ID and status shown.

g. List All Containers

- Displays all containers (running and stopped) using docker ps -a.

h. Stop a Container

- Lets user input a container ID or name.
- Stops the selected container using docker stop.
- Displays success/failure messages.

i. Search Local Docker Image

- Allows users to search locally installed images.
- Displays image info using docker images <name>.

j. Search for Image on DockerHub

- Accepts image name as input.
- Displays search results in a styled QTableWidget:
  - Name, Description, Stars, Official status, Pull Count.
- Automatically generates DockerHub link and shows it in the log.

k. Pull Image from DockerHub

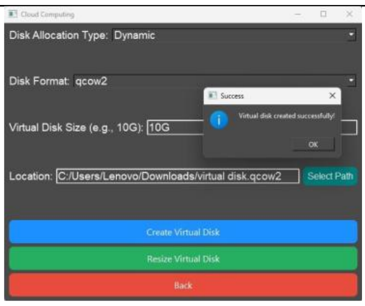
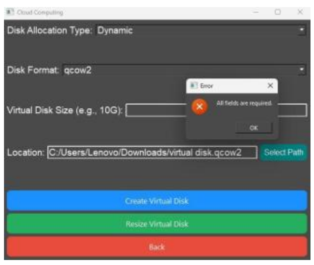
- Lets the user pull a Docker image using docker pull.
- Logs full output to the interface for transparency.

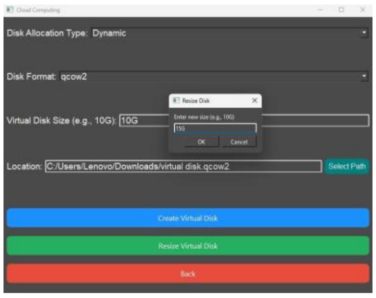
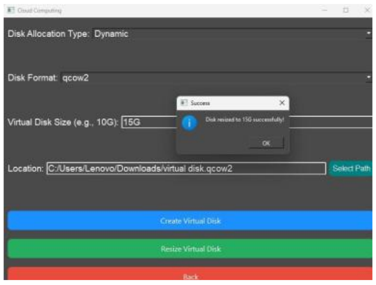
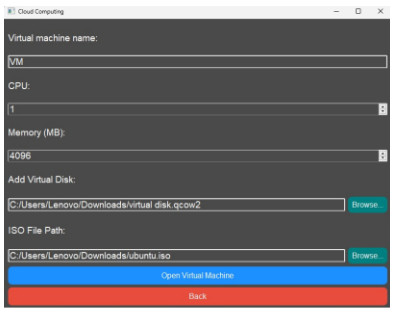
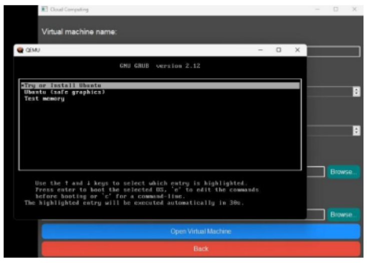
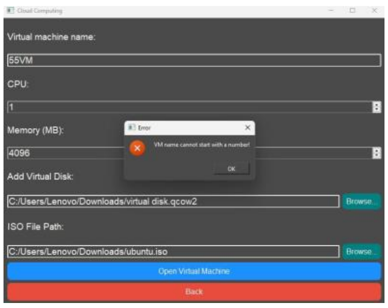
GUI Overview

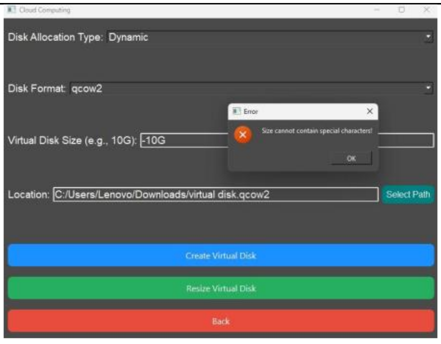
Page	Purpose
Start Menu	Welcome screen, provides access to the project or exit
Project Page	Central page to navigate between VMs, disks, and Docker features
Virtual Disk Manager	Creates, resizes, and selects virtual disk type, size, and format
Virtual Machine Configuration	Configures VM name, CPU, memory, attaches ISO and disk, and launches VM
Docker Tab	Creates Docker files, builds/pulls/searches images, and controls containers
Docker Results	Displays search results from Docker Hub with visual formatting and status

Testing Section

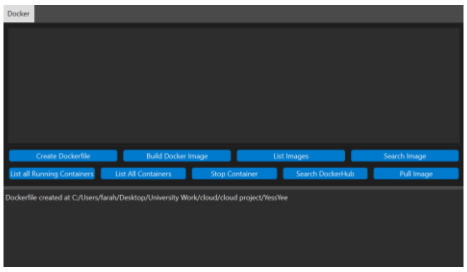
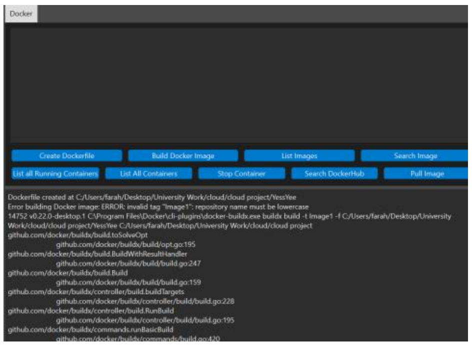
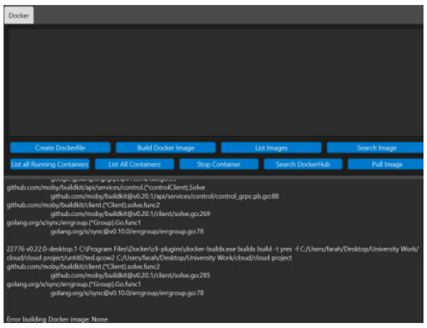
1. Virtual Machine/ Virtual Disk

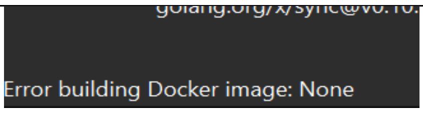
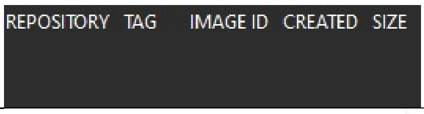
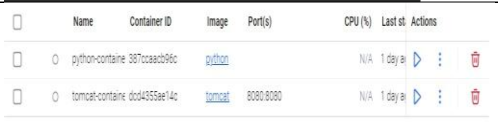
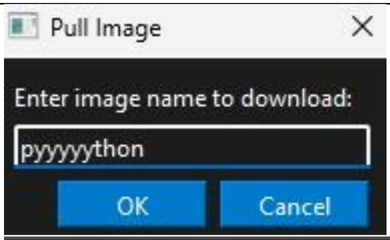
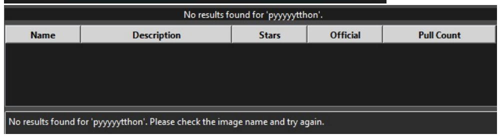
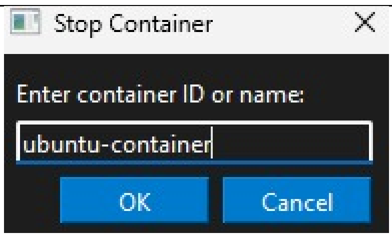
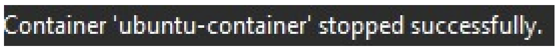
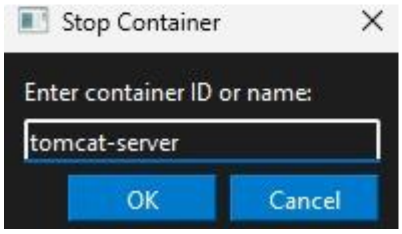
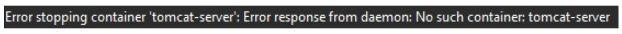
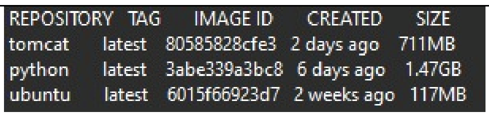
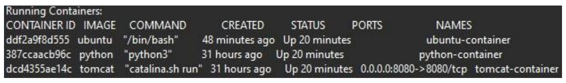
Test Case	Action	Expected Result	Screenshot
Disk Creation – Valid Input	10G qcow2 disk creation	Disk created successfully	 Cloud Computing Disk Allocation Type: Dynamic Disk Format: qcow2 Virtual Disk Size (e.g., 10G): 10G Location: C:/Users/Lenovo/Downloads/virtual disk.qcow2 Success Virtual disk created successfully! OK Select Path Create Virtual Disk Resize Virtual Disk Back
Disk Creation – Missing Size	Attempted creation with no size	Error message displayed	 Cloud Computing Disk Allocation Type: Dynamic Disk Format: qcow2 Virtual Disk Size (e.g., 10G): Location: C:/Users/Lenovo/Downloads/virtual disk.qcow2 Error All fields are required. OK Select Path Create Virtual Disk Resize Virtual Disk Back

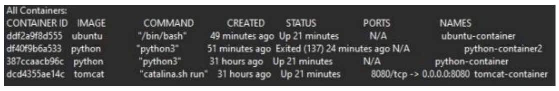
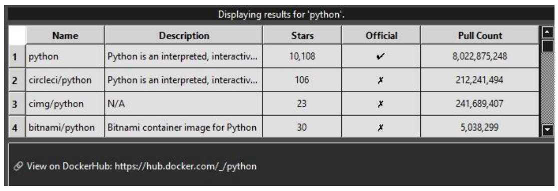
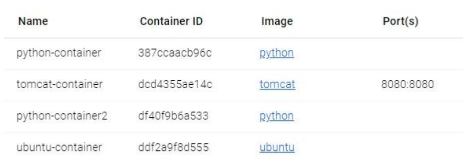
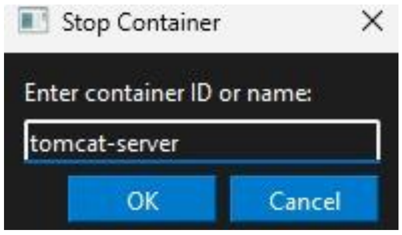
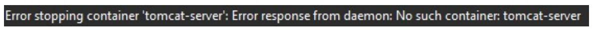
Disk Resize – Expansion	Increased 10G disk to 15G	Disk resized successfully	 Disk Allocation Type: Dynamic Disk Format: qcow2 Virtual Disk Size (e.g., 10G): 10G Resize Disk Enter new size (e.g., 10G) 15G Location: C:/Users/Lenovo/Downloads/virtual disk.qcow2 Cloud Computing OK Cancel Select Path Create Virtual Disk Resize Virtual Disk Back  Disk Allocation Type: Dynamic Disk Format: qcow2 Virtual Disk Size (e.g., 10G): 15G Success Disk resized to 15G successfully! Location: C:/Users/Lenovo/Downloads/virtual disk.qcow2 Cloud Computing OK Select Path Create Virtual Disk Resize Virtual Disk Back
VM Creation – Complete Parameters	Created VM with 2 cores, 4GB RAM, disk, and ISO	VM launched successfully	 Virtual machine name: VM CPU: 1 Memory (MB): 4096 Add Virtual Disk: C:/Users/Lenovo/Downloads/virtual disk.qcow2 ISO File Path: C:/Users/Lenovo/Downloads/ubuntu.iso Cloud Computing Browse... Browse... Open Virtual Machine Back  QEMU GNU GRUB version 2.12 *Try or Install Ubuntu Ubuntu (safe graphics) Test memory Use the ↑ and ↓ keys to select which entry is highlighted. Press enter to boot the selected OS, 'e' to edit the commands before booting or 'c' for a command-line. The highlighted entry will be executed automatically in 30s. Virtual machine name: Cloud Computing Browse... Browse... Open Virtual Machine Back
VM Creation – Name starting with a number	Created a virtual machine with a name starting with a number	Error message displayed	 Virtual machine name: 55VM CPU: 1 Memory (MB): 4096 Error VM name cannot start with a number! Add Virtual Disk: C:/Users/Lenovo/Downloads/virtual disk.qcow2 ISO File Path: C:/Users/Lenovo/Downloads/ubuntu.iso Cloud Computing OK Browse... Browse... Open Virtual Machine Back
Virtual Disk – Special characters in	Added special characters in	Error message displayed	

the size section	the size option		 Disk Allocation Type: Dynamic Disk Format: qcow2 Virtual Disk Size (e.g., 10G): -10G Error: Size cannot contain special characters! Location: C:/Users/Lenovo/Downloads/virtual disk.qcow2 Cloud Computing OK Select Path Create Virtual Disk Resize Virtual Disk Back

2. Docker

Test Case	Action	Expected Result	Screenshot
Dockerfile Creation – Valid Input	Create Dockerfile and save it to valid path	File saved successfully, confirmation log	 Docker Create Dockerfile Build Docker Image List Images Search Image List all Running Containers List All Containers Stop Container Search DockerHub Pull Image Dockerfile created at C:/Users/farah/Desktop/University Work/cloud/cloud project/YessYee
Build Docker Image – Valid Build	Select Dockerfile and image name, click Build	Image built, confirmation in logs	 Docker Create Dockerfile Build Docker Image List Images Search Image List all Running Containers List All Containers Stop Container Search DockerHub Pull Image Dockerfile created at C:/Users/farah/Desktop/University Work/cloud/cloud project/YessYee Error building Docker image: ERROR: invalid tag "Image1": repository name must be lowercase 14752 v0.22.0-desktop.1 C:\Program Files\Docker\cli-plugins\docker-buildx.exe buildx build -t Image1 -f C:/Users/farah/Desktop/University Work/cloud/cloud project/YessYee C:/Users/farah/Desktop/University Work/cloud/cloud project github.com/docker/buildx/build.toSolveOpt github.com/docker/buildx/build/opt.go:195 github.com/docker/buildx/build.BuildWithResultHandler github.com/docker/buildx/build/build.go:247 github.com/docker/buildx/build.Build github.com/docker/buildx/build/build.go:159 github.com/docker/buildx/controller/build.buildTargets github.com/docker/buildx/controller/build/build.go:228 github.com/docker/buildx/controller/build.RunBuild github.com/docker/buildx/controller/build/build.go:195 github.com/docker/buildx/commands.runBasicBuild github.com/docker/buildx/commands/build.go:420
Build Docker Image – Invalid File	Select a non-existent Dockerfile	Error message in logs	 Docker Create Dockerfile Build Docker Image List Images Search Image List all Running Containers List All Containers Stop Container Search DockerHub Pull Image github.com/moby/buildkit/api/services/control.(*controlClient).Solve github.com/moby/buildkit@v0.20.1/api/services/control/control_grpc.pb.go:88 github.com/moby/buildkit/client.(*Client).solve.func2 github.com/moby/buildkit@v0.20.1/client/solve.go:269 golang.org/x/sync/errgroup.(*Group).Go.func1 golang.org/x/sync@v0.10.0/errgroup/errgroup.go:78 22776 v0.22.0-desktop.1 C:\Program Files\Docker\cli-plugins\docker-buildx.exe buildx build -t yres -f C:/Users/farah/Desktop/University Work/cloud/cloud project/untitl2ted.qcow2 C:/Users/farah/Desktop/University Work/cloud/cloud project github.com/moby/buildkit/client.(*Client).solve.func2 github.com/moby/buildkit@v0.20.1/client/solve.go:285 golang.org/x/sync/errgroup.(*Group).Go.func1 golang.org/x/sync@v0.10.0/errgroup/errgroup.go:78 Error building Docker image: None

			 golang.org/x/sync@v0.10. Error building Docker image: None
Show Results of Empty Docker	Select Display all when nothing is created yet	List of all is displayed	 REPOSITORY TAG IMAGE ID CREATED SIZE
Pull Docker Image – Valid Name	Enter known image name (e.g., yess)	Image downloaded successfully	 Name · Container ID · Image · Port(s) · CPU (%) · Last st · Actions python-containe · 387ccaacb96c · python · · N/A · 1 day a tomcat-containe · dcd4355ae14c · tomcat · 8080:8080 · N/A · 1 day a
Pull Docker Image- Invalid Name	Enter an invalid/ not present in the docker name	Error message is displayed	 Pull Image Enter image name to download: pyyyyython  No results found for 'pyyyyytthon'. Name · Description · Stars · Official · Pull Count No results found for 'pyyyyytthon'. Please check the image name and try again.
Stop Container – Valid ID	Enter ID of running container	Container stops, message shown	 Stop Container Enter container ID or name: ubuntu-container  Container 'ubuntu-container' stopped successfully.
Stop Container- Invalid ID	Enter ID that is not listed before	Error message is displayed	 Stop Container Enter container ID or name: tomcat-server  Error stopping container 'tomcat-server': Error response from daemon: No such container: tomcat-server
List Images from Docker	Click on the option List Images	Displays all images	 REPOSITORY TAG IMAGE ID CREATED SIZE tomcat latest 80585828cfe3 2 days ago 711MB python latest 3abe339a3bc8 6 days ago 1.47GB ubuntu latest 6015f66923d7 2 weeks ago 117MB
List all running Containers	Click on the option All running Containers	Displays all running containers	 Running Containers: CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES ddf2a9f8d555 ubuntu "/bin/bash" 48 minutes ago Up 20 minutes ubuntu-container 387ccaacb96c python "python3" 31 hours ago Up 20 minutes python-container dcd4355ae14c tomcat "catalina.sh run" 31 hours ago Up 20 minutes 0.0.0.0:8080->8080/tcp tomcat-container

List All Available Containers	Click on the Option All Containers	Display all available Containers	
List All Dockers on Desktop	Click on option show all Dockers	Displays all Dockers available on the desktop	 
Stop Container-Invalid Name	Click on Search and enter the wrong name	Displays an error message	 

```
All Containers:
CONTAINER ID   IMAGE      COMMAND          CREATED          STATUS                        PORTS                     NAMES
ddf2a9f8d555   ubuntu     "/bin/bash"      49 minutes ago   Up 21 minutes                 N/A                       ubuntu-container
df40f9b6a533   python     "python3"        51 minutes ago   Exited (137) 24 minutes ago   N/A                       python-container2
387ccaacb96c   python     "python3"        31 hours ago     Up 21 minutes                 N/A                       python-container
dcd4355ae14c   tomcat     "catalina.sh run"  31 hours ago   Up 21 minutes                 8080/tcp -> 0.0.0.0:8080  tomcat-container
```

Displaying results for 'python'.

	Name	Description	Stars	Official	Pull Count
1	python	Python is an interpreted, interactiv...	10,108	✔	8,022,875,248
2	circleci/python	Python is an interpreted, interactiv...	106	✗	212,241,494
3	cimg/python	N/A	23	✗	241,689,407
4	bitnami/python	Bitnami container image for Python	30	✗	5,038,299

View on DockerHub: https://hub.docker.com/_/python

Name	Container ID	Image	Port(s)
python-container	387ccaacb96c	python	
tomcat-container	dcd4355ae14c	tomcat	8080:8080
python-container2	df40f9b6a533	python	
ubuntu-container	ddf2a9f8d555	ubuntu	

Error stopping container 'tomcat-server': Error response from daemon: No such container: tomcat-server

Challenges & Solutions

•Challenges Faced and Solutions for Virtual Disk Creation:

1.Incorrect or Missing Disk Path

➔Challenge: Users might not provide a valid or correctly formatted file path.

➔Solution: GUI uses a file dialog (QFileDialog) to guide path selection.

2.Invalid Disk Size Input

➔Challenge: Users might enter a size without units or with special characters.

➔Solution: Size is validated to include a unit (G, M, or T).
Custom logic checks for and blocks special characters using special_chars.
Displays an error dialog (QMessageBox) if validation fails.

•Challenges Faced and Solutions for Virtual Machine Creation:

1. Validating User Inputs for VM Configuration

➔Challenge: Users might enter invalid VM names (e.g., starting with digits or including special characters), invalid memory sizes, or forget to provide ISO/virtual disk paths.

➔Solution: Implemented robust input validation checks:
Disallowed VM names starting with digits or containing special characters.
Checked for required fields like ISO and disk path before proceeding.
Provided real-time user feedback via QMessageBox for better usability.

2. Supporting User-Friendly File Selection

➔Challenge: Manually typing ISO or virtual disk paths is error-prone.

➔Solution:
Implemented file browsers using QFileDialog.getOpenFileName() for both ISO and disk selection, making it user-friendly and reducing file path errors.

3. Error Handling for VM Launch Failures

➔Challenge: If QEMU fails to start (due to missing files or permission issues), it wasn't clear to users why it failed.

➔Solution: Wrapped the QEMU launch command in a try-except block and showed the exception message via a message box to assist in debugging: except Exception as e:QMessageBox.critical(self, "Error", f"Failed to launch VM:\n{e}")

•Challenges Faced and Solutions for Docker Management:

1. Invalid or Empty Dockerfile Content

➔ Challenge: Users may attempt to create a Dockerfile with empty or invalid content.

➔ Solution: Provided a QTextEdit field for preview/editing and validation. Showed an error message if content is empty or malformed before saving.

2. Docker Not Installed or Not Running

➔ Challenge: Docker commands fail if Docker is not installed or the daemon is not running.

➔ Solution: Used try-except blocks to catch FileNotFoundError and notify the user clearly: "Ensure Docker is installed and running."

3. Build Failures with Unclear Error Messages

➔ Challenge: Errors during docker build could be hard to understand.

➔ Solution: Captured and displayed detailed stderr output using capture_output=True to help users troubleshoot errors.

4. Ambiguous Image Name During Search or Pull

➔ Challenge: Users might enter an incorrect or partial image name.

➔ Solution: Implemented a DockerHub search tool with formatted output (name, description, stars, etc.) and validation of the image name.

5. Pulling Large Images without Progress Feedback

➔ Challenge: Downloading large images could make users think the system is frozen.

➔ Solution: Added a status label (status_label.setText()) to indicate that a pull is in progress, improving UX.

6. Overwriting Existing Dockerfiles

➔ Challenge: Saving a Dockerfile to an existing path could overwrite important content.

➔ Solution: Used QFileDialog to prompt the user and confirm overwrite behavior.

7. Container Stop Errors for Invalid IDs

➔ Challenge: Stopping a container using an incorrect ID or name results in silent failure.

➔ Solution: Displayed detailed error messages if stopping fails and advised the user to check the container ID.

User Manual

System Requirements

- **Operating System**: Windows 10 or 11 (64-bit)
- **Python Version**: 3.8 or higher
- **Dependencies**:
  - PySide6
  - requests
- **Tools Required**:
  - QEMU (Ensure path to qemu-img.exe and qemu-system-x86_64.exe is correct)
  - Docker (Must be installed and running)
- **QEMU Executables Location:**
  - C:/msys64/ucrt64/bin/qemu-img.exe
  - C:/msys64/ucrt64/bin/qemu-system-x86_64.exe

Virtual Disk Management

➔ Create Virtual Disk

1. Select disk **format** (e.g., qcow2, raw, vmdk, vdi, etc.).
2. Choose **allocation type**: Dynamic or Fixed.
3. Enter **disk size** (e.g., 10G, 1024M).
4. Select a file path to save the disk.
5. Click **Create Virtual Disk**.

➔ Resize Virtual Disk

1. Click **Resize Virtual Disk**.
2. Choose an existing disk file.
3. Enter a **larger size**.
4. Confirm. The disk will expand.

Virtual Machine Configuration

➔ Create Virtual Machine

1. Enter VM **name** (no special characters or digits at start).
2. Select number of **CPU cores** (1–4).
3. Set **memory** (512 MB to 32 GB).
4. Select a **virtual disk** created earlier.
5. Browse and choose an **ISO file**.
6. Click **Open Virtual Machine** to launch.
7. A new QEMU window will open with the VM running.

Docker Management (via Docker Tab)

➔ Create Docker file

1. Write or paste Docker file content into the text area.
2. Click **Create Docker file**.
3. Choose a save location and confirm.

➔ Build Docker Image

1. Click **Build Docker Image**.
2. Select the Docker file.
3. Enter an image name.
4. The image will be built and logs shown in the output area.

➔ List Docker Images

Click **List Images** to show all installed Docker images.

➔ Search Installed Image

Click **Search Image**, enter a name (e.g., ubuntu), and view local image details.

➔ Search DockerHub

1. Click **Search DockerHub**.
2. Enter image name (ex.ImageYess).
3. Results will appear in a table with:
   - Name
   - Description
   - Stars
   - Official status
   - Pull count
4. A clickable link to DockerHub is also provided.

➔ Pull Image from DockerHub

Click **Pull Image**, enter image name (e.g., python:3.10), and wait for it to download.

➔ List Running Containers

Click **List all Running Containers** to view active containers.

➔ List All Containers

Click **List All Containers** to see running and stopped containers.

➔ Stop a Container

1. Click **Stop Container**.

2. Enter container **ID or name**.
3. The container will be stopped and confirmed in the log.

Conclusion

This project successfully integrated two major virtualization tools—QEMU and Docker—into one centralized management system. Through interactive GUI development, robust validation, and external system control, we created a powerful educational tool for students learning about cloud environments.

The Cloud Management System demonstrates how modern virtualization tasks can be abstracted into simple workflows, making it accessible for users with minimal technical background. It offers a foundation that can be extended with future features like VM snapshots, container metrics, or remote Docker daemon support.