VirtuManager

### Virtual Disk and Virtual Machine Management Tool

Developed by:

Enjy Ramadan

Mariam Kandeel

Mazen Ashraf

Sohaila Ashraf

Omar Sherif

[Project Demo](https://drive.google.com/file/d/17Cv0_JQ_ZSzXwLnH25qcxVEetLZCE87e/view?usp=sharing)

## Project Overview:

**Overview:**  
VirtuManager is a Python-based graphical application designed for creating and managing virtual disks and virtual machines. It ensures optimal system configuration through validation checks for available CPU, RAM, and disk space before creating virtual machines (VMs). This project marks the first phase of a broader virtualization and container management tool, laying the groundwork for future enhancements like Docker container management.

**Core Features (Phase 1):**

* Creation of virtual disks with customizable settings (name, path, format, size).
* Virtual machine creation with specified CPU, RAM, disk, and ISO image configurations.
* System validation to check available resources before VM creation (CPU cores, RAM, disk space).
* Support for a variety of virtual disk formats (qcow2, vmdk, vdi, raw, etc.).

## Implemented Features:

* **Virtual Disk Creation:** Allows users to create virtual disks with specified parameters such as name, path, format, and size.
* **Virtual Machine Creation:** Facilitates creating VMs by selecting the required CPU, RAM, disk, and ISO image.
* **System Resource Validation:** Ensures that the system has sufficient CPU, RAM, and disk space before VM creation to prevent resource shortages.
* **Support for Multiple Disk Formats:** Includes various virtual disk formats like qcow2, vmdk, vdi, raw, vhd, vhdx, and more, enabling flexibility in virtual machine setups.

## Test Casses:

|  |  |
| --- | --- |
| *Test Case Name* | TVD1: Verify Page Loads Successfully |
| *Test Scenario* | Ensure the Create Virtual Disk page opens correctly. |
| *Prerequisites* |  |
| *Test Input* | Open the Create Disk page |
| *Execution Steps* | 1. Launch the app. 2. Click on “START VD" |
| *Expected Behavior* | The Create Virtual Disk form loads without errors |
| *Assumptions* | UI and navigation work |
| *Actual Results* | UI and navigation work |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVD2: Submit with All Valid Data |
| *Test Scenario* | It should create a disk successfully. |
| *Prerequisites* | Virtualization engine functional |
| *Test Input* | 1. Valid path 2. Valid format 3. Valid size |
| *Execution Steps* | 1. Fill all fields correctly 2. Click “Create Disk” |
| *Expected Behavior* | Disk is created |
| *Assumptions* | Backend supports operation |
| *Actual Results* | Disk is created successfully and ready to function |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVD3: Submit with Empty Fields |
| *Test Scenario* | Submit button should validate empty input. |
| *Prerequisites* | The form is visible |
| *Test Input* | Leave all fields blank |
| *Execution Steps* | 1. Click “Create Disk” without filling anything |
| *Expected Behavior* | Error shown or disk not created |
| *Assumptions* | Validation is present |
| *Actual Results* | Validation is present |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVD4: Verify Active States of Buttons |
| *Test Scenario* | Ensure that all buttons show appropriate visual feedback when clicked. |
| *Prerequisites* | The "Create Virtual Disk" page is fully loaded |
| *Test Input* | Interaction clicks with all buttons |
| *Execution Steps* | 1. Hover and click on the “Browse” button. 2. Hover and click on the “Back” button. 3. Hover and click on the “Create Disk” button. |
| *Expected Behavior* | Each button should visually be active |
| *Assumptions* | UI framework handles button states |
| *Actual Results* | Functioning buttons |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVD5: Verify “Disk Path” Browse Button Functionality |
| *Test Scenario* | Clicking “Browse” opens laptop search |
| *Prerequisites* | The form is visible |
| *Test Input* | Click “Browse” |
| *Execution Steps* | 1. Click on “Browse” |
| *Expected Behavior* | File dialog opens for path selection |
| *Assumptions* | OS-level file dialog is supported |
| *Actual Results* | OS-level file dialog is supported |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVD6: Verify Disk Format Dropdown Options |
| *Test Scenario* | Dropdown should list available formats. |
| *Prerequisites* | The form is visible |
| *Test Input* | Click on the dropdown |
| *Execution Steps* | 1. Click the “Disk Format” dropdown |
| *Expected Behavior* | Shows valid formats like qcow2, vdi, vmdk, raw |
| *Assumptions* | Options are hardcoded or loaded correctly |
| *Actual Results* | Options are hardcoded or loaded correctly |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVD7: Verify Disk Size Spinner Limits |
| *Test Scenario* | Only valid sizes can be selected. |
| *Prerequisites* | The form is visible |
| *Test Input* | Use arrows to change size |
| *Execution Steps* | 1. Adjust disk size |
| *Expected Behavior* | Only positive integers allowed |
| *Assumptions* | Spinner has limits set |
| *Actual Results* | Spinner has limits set |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVD8: Verify Back Button Functionality |
| *Test Scenario* | Clicking “Back” returns to the previous page. |
| *Prerequisites* | Page is open |
| *Test Input* | Click Back |
| *Execution Steps* | 1. Click “Back” |
| *Expected Behavior* | Redirects back |
| *Assumptions* | Navigation handler works |
| *Actual Results* | Goes back to the previous page |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer game  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVD9: Submit with Only Disk Path Filled |
| *Test Scenario* | Incomplete submission should be rejected. |
| *Prerequisites* | The form is visible |
| *Test Input* | Only browse path |
| *Execution Steps* | 1. Enter disk path 2. Click “Create Disk” |
| *Expected Behavior* | Error due to missing fields |
| *Assumptions* | Form requires all fields |
| *Actual Results* | Error message box pops up informing user about missing fields |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVD10: Disk Size Field with Zero |
| *Test Scenario* | Zero GB should not be allowed. |
| *Prerequisites* | The form is visible |
| *Test Input* | 0 in Disk Size |
| *Execution Steps* | 1. Set size to 0 2. Submit |
| *Expected Behavior* | Validation error |
| *Assumptions* | Size must be >0 |
| *Actual Results* | Message box pops up informing the user about the disk size error |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVD11: Disk Format Not Selected |
| *Test Scenario* | Must require a format. |
| *Prerequisites* | Other fields filled |
| *Test Input* | Leave format blank |
| *Execution Steps* | 1. Enter path and size 2. Leave format 3. Submit |
| *Expected Behavior* | Validation error |
| *Assumptions* | Size must be > 0 |
| *Actual Results* | Message box pops up informing the user about the missing information error |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVD12: Disk Already Exists at Path |
| *Test Scenario* | Duplicate disk path should raise an error. |
| *Prerequisites* | File already exists at given path |
| *Test Input* | Use the same path as previous disk |
| *Execution Steps* | 1. Enter existing path 2. Submit form |
| *Expected Behavior* | “File already exists” error |
| *Assumptions* | System checks for file existence |
| *Actual Results* | Message box pops up informing the user about the existing file |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVM1: Verify Page Load |
| *Test Scenario* | Ensure the "Create Virtual Machine" page loads successfully. |
| *Prerequisites* |  |
| *Test Input* | Open the "Create VM" tab |
| *Execution Steps* | 1. Launch the app. 2. Click on “START VM" |
| *Expected Behavior* | The Create Virtual Machine form loads without errors |
| *Assumptions* | UI and navigation work |
| *Actual Results* | UI and navigation work |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVM2: Submit with All Valid Data |
| *Test Scenario* | It should create a machine successfully. |
| *Prerequisites* | Virtualization engine functional |
| *Test Input* | 1. Disk input 2. Memory (GB) 3. Number of CPUs 4. ISO file |
| *Execution Steps* | 1. Fill all fields correctly 2. Click “Create Machine” |
| *Expected Behavior* | Machine is created |
| *Assumptions* | Backend supports operation |
| *Actual Results* | Machine is created successfully and ready to function |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVM3: Submit with Empty Fields |
| *Test Scenario* | Submit button should validate empty input. |
| *Prerequisites* | The Form is visible |
| *Test Input* | Leave all fields blank |
| *Execution Steps* | 1. Click “Create Machine” without filling anything |
| *Expected Behavior* | Error shown or machine not created |
| *Assumptions* | Validation is present |
| *Actual Results* | Validation is present |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVM4: Validate Feedback on VM Creation Failure |
| *Test Scenario* | Show clear message if VM creation fails (e.g., missing folder or insufficient disk). |
| *Prerequisites* | Known failure scenario exists (like in image: not enough space) |
| *Test Input* | Set disk to invalid path  ” D:/Games/DeadByDaylight/Manifest\_NonUFSFiles\_EGS.txt” |
| *Execution Steps* | 1. Fill form with faulty disk path. 2. Click "Create VM". |
| *Expected Behavior* | An error message box pops up informing the user. |
| *Assumptions* | Errors returned from backend |
| *Actual Results* | Errors returned from backend |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVM5: Verify Back Button Functionality |
| *Test Scenario* | Clicking “Back” returns to the previous page. |
| *Prerequisites* | Page is open |
| *Test Input* | Click Back |
| *Execution Steps* | 1. Click “Back” |
| *Expected Behavior* | Redirects back |
| *Assumptions* | Navigation handler works |
| *Actual Results* | Goes back to the previous page |
| *Status* | Pass |
| *Real Life Testing* | A computer screen shot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVM6: Validate Memory Field (Positive Integer) |
| *Test Scenario* | Ensure the user can only enter a positive integer for memory. |
| *Prerequisites* | Input box for Memory is active |
| *Test Input* | “1” |
| *Execution Steps* | 1. Click on the Memory field. 2. Enter a value like 1. |
| *Expected Behavior* | Field accepts the number and stores it as an integer |
| *Assumptions* | Validation is present |
| *Actual Results* | Validation is present |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVM7: Validate Memory Field (Invalid Input) |
| *Test Scenario* | Ensure the field rejects negative values. |
| *Prerequisites* | Memory field input not restricted by default |
| *Test Input* | “-1” |
| *Execution Steps* | 1. Click Memory field. 2. Enter invalid input. |
| *Expected Behavior* | Field blocks invalid input or displays error |
| *Assumptions* | Validation is present |
| *Actual Results* | Validation is present |
| *Status* | Pass |
| *Real Life Testing* |  |

|  |  |
| --- | --- |
| *Test Case Name* | TVM8: Validate CPUs Field (Positive Integer) |
| *Test Scenario* | Ensure the user can only enter valid CPU core counts. |
| *Prerequisites* | The CPU field is editable |
| *Test Input* | “1” |
| *Execution Steps* | 1. Enter value in CPUs field. |
| *Expected Behavior* | Field accepts integer values |
| *Assumptions* | Validation is present |
| *Actual Results* | Validation is present |
| *Status* | Pass |
| *Real Life Testing* | A screenshot of a computer  AI-generated content may be incorrect. |

|  |  |
| --- | --- |
| *Test Case Name* | TVM9: Validate CPUs Field (Negative Integer) |
| *Test Scenario* | Ensure the user can only enter valid CPU core counts. |
| *Prerequisites* | The CPU field is editable |
| *Test Input* | “-1” |
| *Execution Steps* | 1. Enter value in CPUs field. |
| *Expected Behavior* | Field does not accept negative integer values |
| *Assumptions* | Validation is present |
| *Actual Results* | Validation is present |
| *Status* | Pass |
| *Real Life Testing* |  |

|  |  |
| --- | --- |
| *Test Case Name* | TVM10: Verify Memory and CPU Limits |
| *Test Scenario* | Ensure application restricts memory and CPU input to realistic values. |
| *Prerequisites* | Limits defined in code |
| *Test Input* | Enter extreme values Memory= “9999”, CPU= “128” |
| *Execution Steps* | 1. Enter large numbers in both fields. |
| *Expected Behavior* | Error message is shown |
| *Assumptions* | Validation is present |
| *Actual Results* | Validation is present |
| *Status* | Pass |
| *Real Life Testing* |  |

|  |  |
| --- | --- |
| *Test Case Name* | TVM11: Verify Minimum Memory/CPU Requirement |
| *Test Scenario* | Prevent VM creation if memory/CPU value is below minimum allowed. |
| *Prerequisites* | Backend enforces min values. |
| *Test Input* | Memory: 0 / CPU: 0 |
| *Execution Steps* | 1. Set memory and CPU values to 0. 2. Click Create VM. |
| *Expected Behavior* | Error message is shown |
| *Assumptions* | Error message informing the user about the incorrect value |
| *Actual Results* | Error message informing the user about the incorrect value |
| *Status* | Pass |
| *Real Life Testing* |  |

|  |  |
| --- | --- |
| *Test Case Name* | TVM12: Validate ISO Path Selection |
| *Test Scenario* | Ensure users can select ISO file. |
| *Prerequisites* | ISO files are available in file system |
| *Test Input* | Click "Browse" |
| *Execution Steps* | 1. Click "Browse" next to ISO Path. 2. Select the ISO file. |
| *Expected Behavior* | ISO Path field updates with file location |
| *Assumptions* | ISO file format supported |
| *Actual Results* | ISO file format supported |
| *Status* | Pass |
| *Real Life Testing* |  |

|  |  |
| --- | --- |
| *Test Case Name* | TVM13: Verify ISO Path is Optional |
| *Test Scenario* | Check behavior when ISO path is left blank. |
| *Prerequisites* | ISO optional |
| *Test Input* | Leave ISO empty, fill rest |
| *Execution Steps* | 1. Fill required fields. 2. Leave ISO blank. 3. Click Create VM. |
| *Expected Behavior* | VM creation succeeds without ISO |
| *Assumptions* | Validation is present |
| *Actual Results* | Validation is present |
| *Status* | Pass |
| *Real Life Testing* |  |

|  |  |
| --- | --- |
| *Test Case Name* | TVM14: Validate File Format of ISO |
| *Test Scenario* | Only .iso files are allowed. |
| *Prerequisites* | File system has non-ISO files. |
| *Test Input* | Select non-ISO file. |
| *Execution Steps* | 1. Click Browse. 2. Try selecting .txt or .exe file. |
| *Expected Behavior* | System restricts or warns user |
| *Assumptions* | Validation is present |
| *Actual Results* | Validation is present |
| *Status* | Pass |
| *Real Life Testing* |  |

|  |  |
| --- | --- |
| *Test Case Name* | THP1: Verify Page Load |
| *Test Scenario* | Ensure the "HOME" page loads successfully. |
| *Prerequisites* |  |
| *Test Input* |  |
| *Execution Steps* | 1. Launch the app. |
| *Expected Behavior* | The home page loads without errors |
| *Assumptions* | UI and navigation work |
| *Actual Results* | UI and navigation work |
| *Status* | Pass |
| *Real Life Testing* |  |

|  |  |
| --- | --- |
| *Test Case Name* | THP2: Verify “Start VD” button |
| *Test Scenario* | Ensure the “Start VD” button loads the “virtual disk page” correctly. |
| *Prerequisites* |  |
| *Test Input* | Button input |
| *Execution Steps* | 1. Press on start VD |
| *Expected Behavior* | The “virtual disk” page loads without errors |
| *Assumptions* | Virtual disk page is loaded |
| *Actual Results* | Virtual disk page is loaded |
| *Status* | Pass |
| *Real Life Testing* |  |

|  |  |
| --- | --- |
| *Test Case Name* | THP3: Verify “Start VM” button |
| *Test Scenario* | Ensure the “Start VM” button loads the “virtual machine page” correctly. |
| *Prerequisites* |  |
| *Test Input* | Button input |
| *Execution Steps* | 1. Press on start VM |
| *Expected Behavior* | The “virtual machine” page loads without errors |
| *Assumptions* | Virtual machine page is loaded |
| *Actual Results* | Virtual machine page is loaded |
| *Status* | Pass |
| *Real Life Testing* |  |

|  |  |
| --- | --- |
| *Test Case Name* | THP4: Verify the “Watch” button for the VM |
| *Test Scenario* | Ensure the “watch” button loads the “YouTube page” correctly. |
| *Prerequisites* |  |
| *Test Input* | Button input |
| *Execution Steps* | 1. Press on “watch” |
| *Expected Behavior* | The “YouTube” page loads without errors |
| *Assumptions* | YouTube page is loaded |
| *Actual Results* | YouTube page is loaded |
| *Status* | Pass |
| *Real Life Testing* |  |

|  |  |
| --- | --- |
| *Test Case Name* | THP5: Verify the “Watch” button for the VD |
| *Test Scenario* | Ensure the “watch” button loads the “YouTube page” correctly. |
| *Prerequisites* |  |
| *Test Input* | Button input |
| *Execution Steps* | 1. Press on “watch” |
| *Expected Behavior* | The “YouTube” page loads without errors |
| *Assumptions* | YouTube page is loaded |
| *Actual Results* | YouTube page is loaded |
| *Status* | Pass |
| *Real Life Testing* |  |

## Testing:

**Testing Approach:**

* **Unit Testing:** Each module, including disk and VM creation, was unit tested to ensure accuracy in functionality.
* **System Validation Testing:** Ensured that the system resource checks (CPU, RAM, disk) work accurately before creating VMs.
* **User Interface Testing:** Verified the Tkinter GUI for proper user interaction, ensuring inputs are captured correctly and VM/disk creation flows smoothly.
* **Compatibility Testing:** Tested the tool with different virtual disk formats (qcow2, vmdk, etc.) to verify format compatibility.

**Key Results:**

* System resource validation: Ensured no VM creation occurred without sufficient resources.
* Multiple disk format handling: Ensured compatibility across formats without failure.
* GUI interaction: User inputs for VM and disk creation were processed seamlessly.

## User Manual:

**How to Use VirtuManager**

1. **Installation:**
   * Install dependencies using the following command:
2. **Create Virtual Disk:**
   * Launch VirtuManager.
   * Enter the required parameters (disk name, path, format, and size).
   * Click on "Create Disk" to generate the virtual disk.
3. **Create Virtual Machine:**
   * In the GUI, the CPU, RAM, disk, and ISO image.
   * Click "Create VM." The system will validate available resources before proceeding.
   * If resources are insufficient, a warning will appear; otherwise, the VM will be created.
4. **Running the Application:**
   * Open the terminal and run the VirtuManager application.
   * The GUI will appear, allowing interaction with the disk and VM creation features.

**Troubleshooting:**

* **Error: Insufficient Resources**
  + Make sure your system has enough CPU cores, RAM, and disk space.
  + If validation fails, close unnecessary applications and try again.
* **Disk Format Issues:**
  + Ensure you're selecting a supported disk format (qcow2, vmdk, etc.).
  + If an unsupported format is selected, the tool will show an error message.