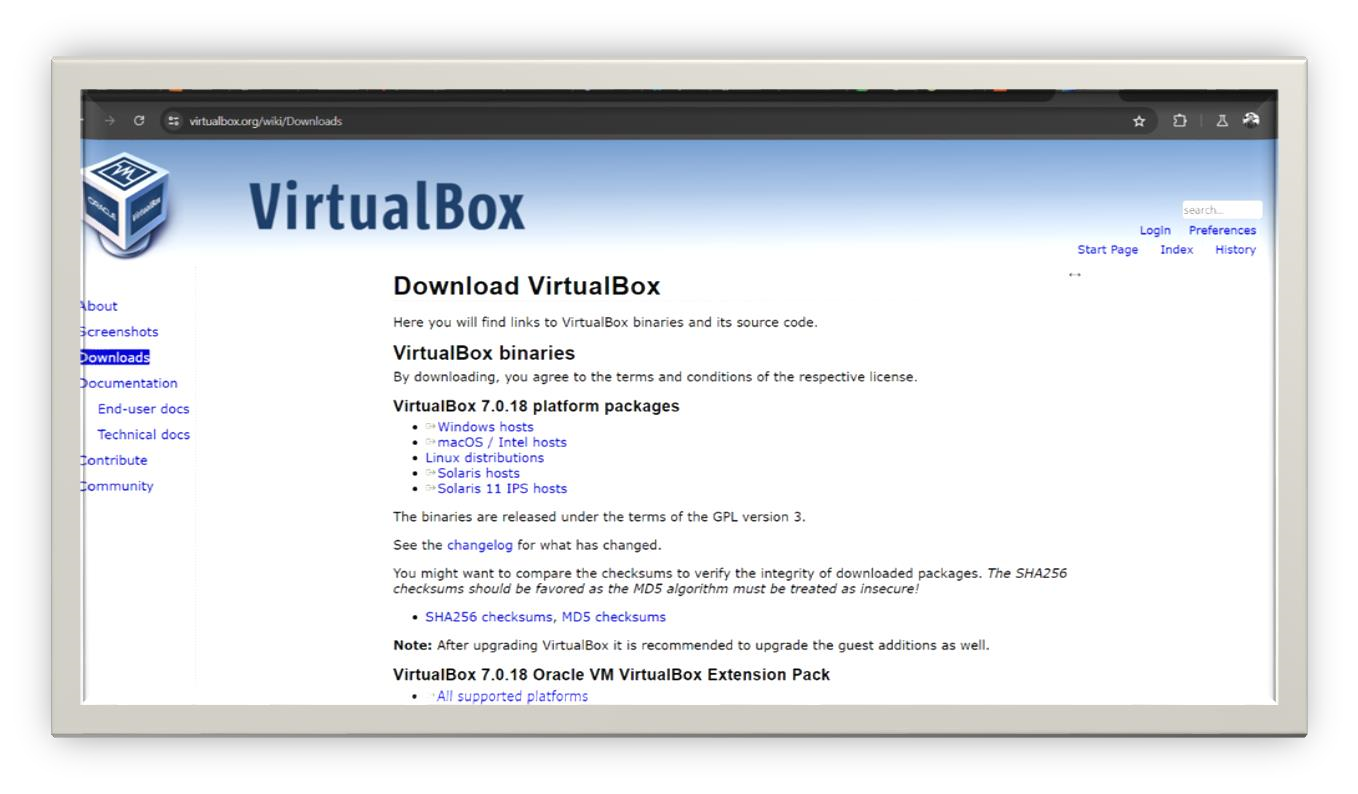
**Task Q3**



* Installing ubuntu ISO <https://ubuntu.com/download/desktop>

• Installing virtualbox 

Name: Angadi Saiganesh Batch: B5

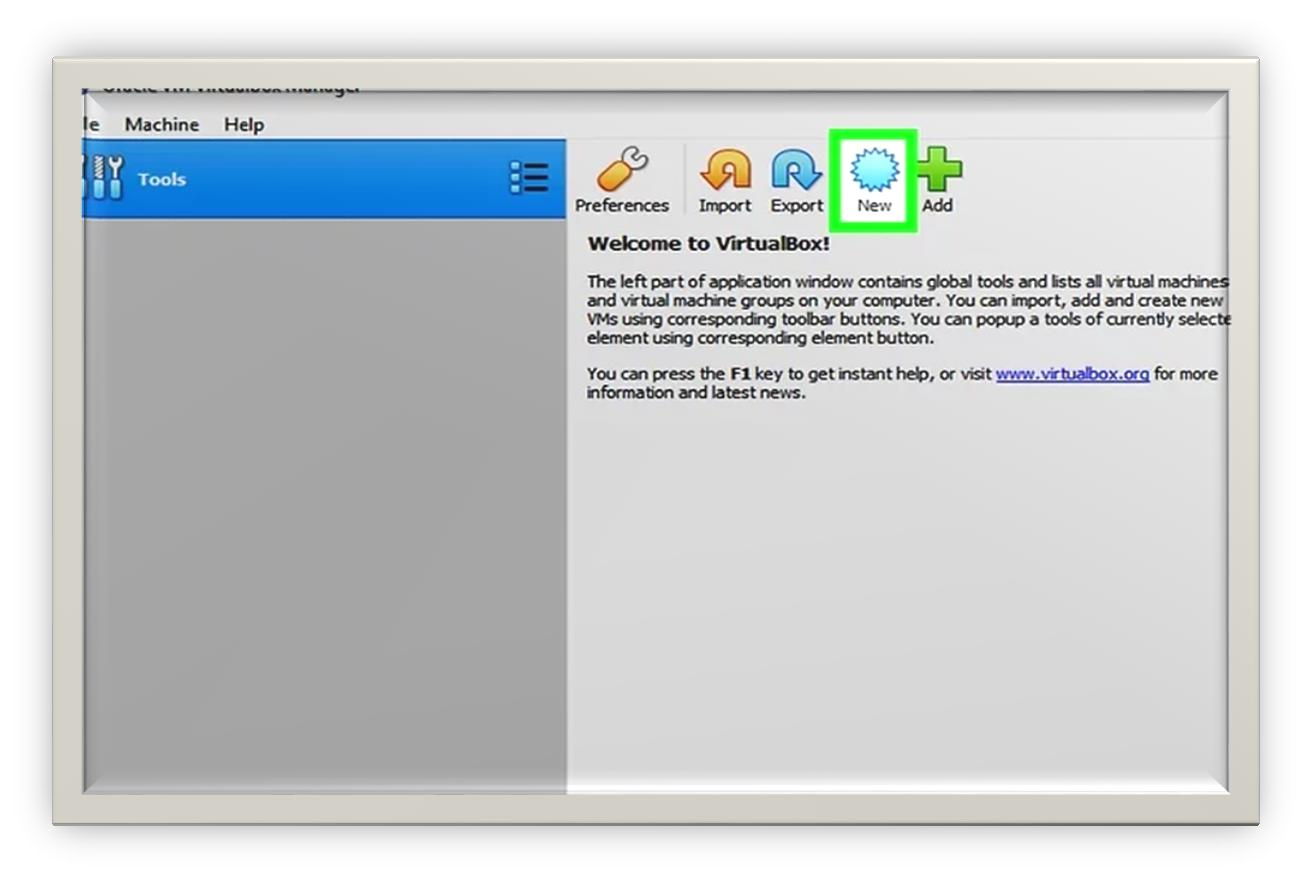
https://www.virtualbox.org/wiki/Downloads



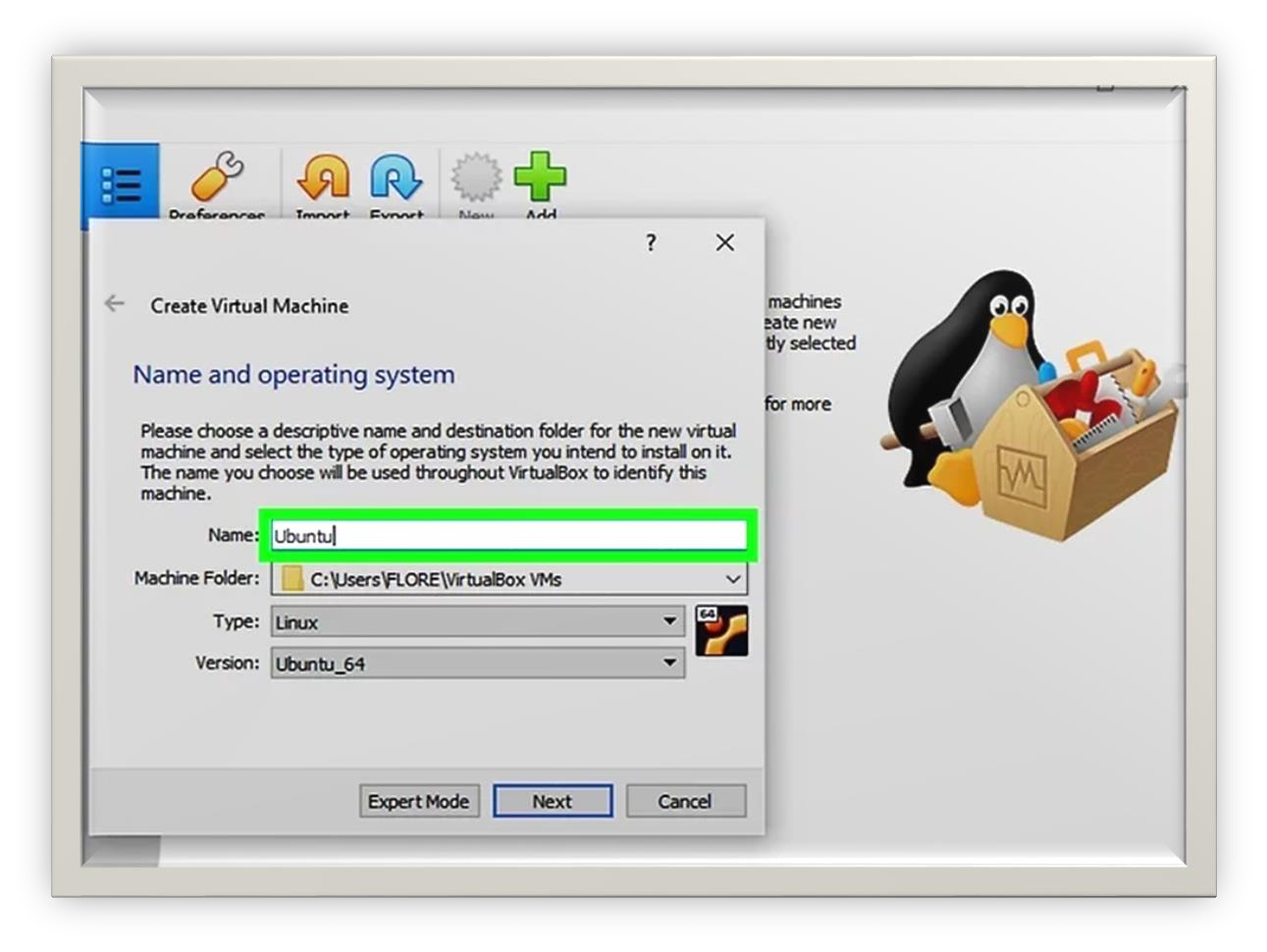
•

O

pen virtual box



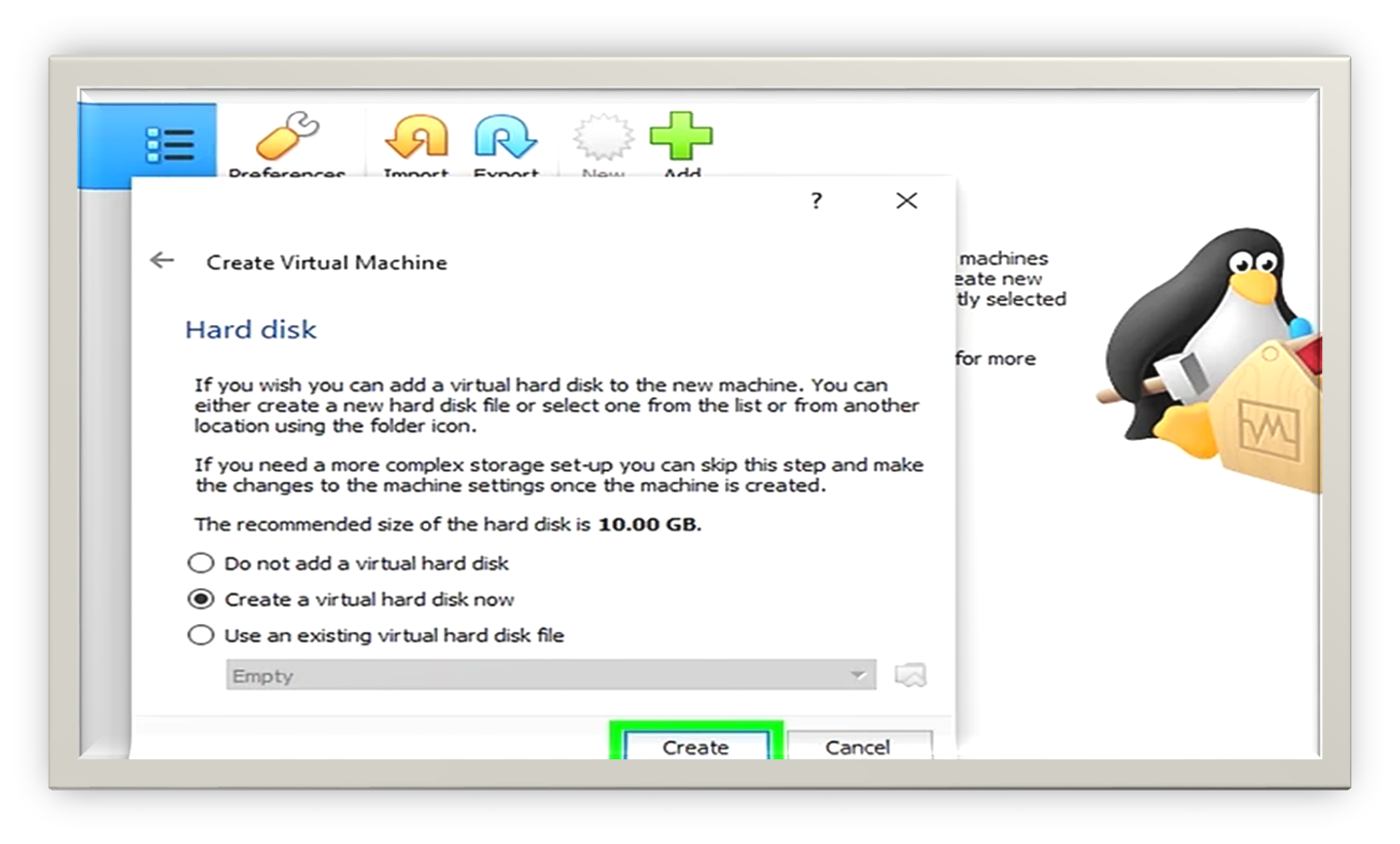
* Click on new and fill the basic data



Name: Angadi Saiganesh Batch: B5



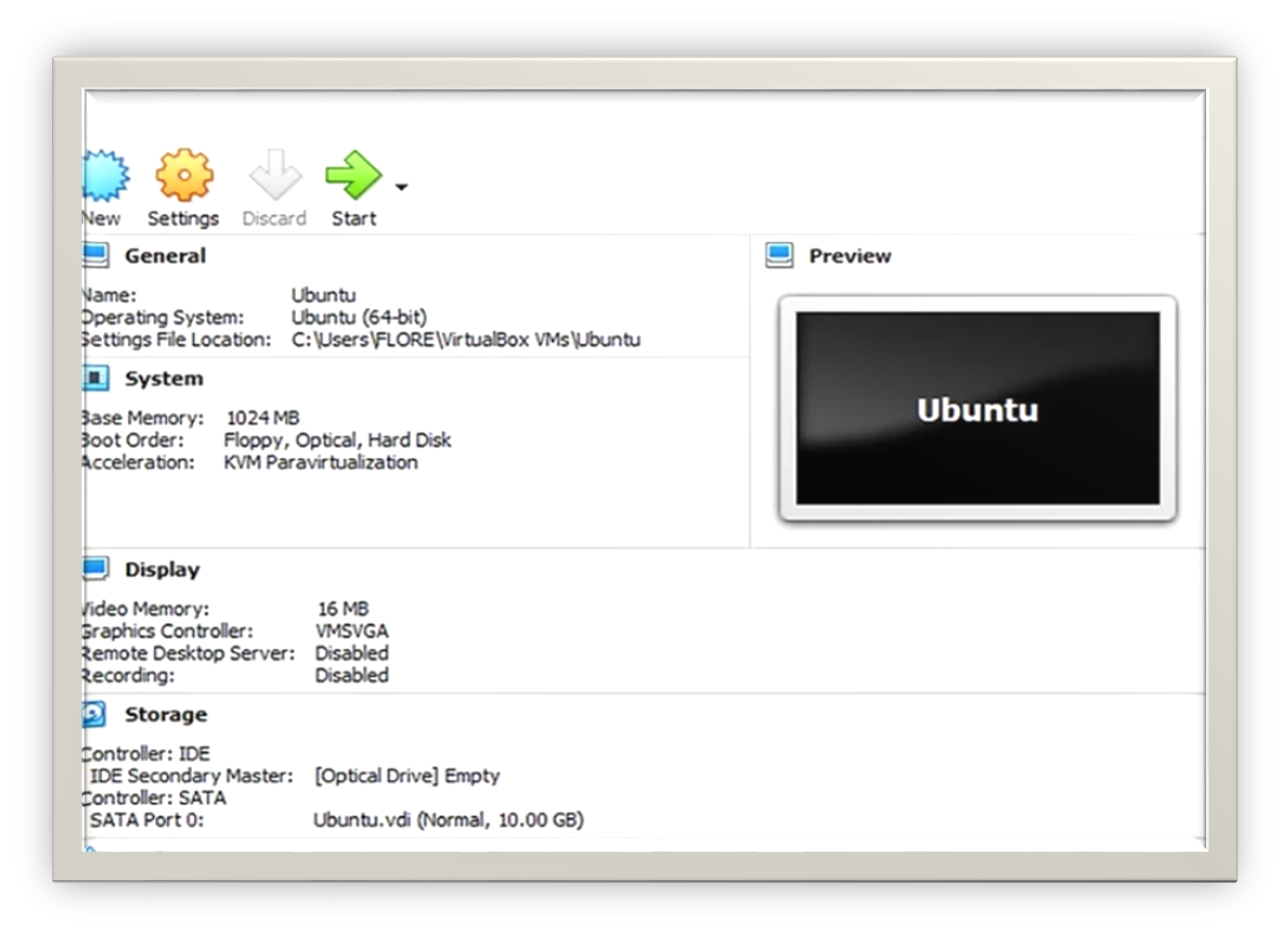
* Set memory size favorable and depending up your host recommended to have 2GB of ram atleast
* Set Vdisk size (virtual disk) The storage is recommended to have 20 GB normally for linux I suggest you to have size of 32 GB atleast
* Click on start



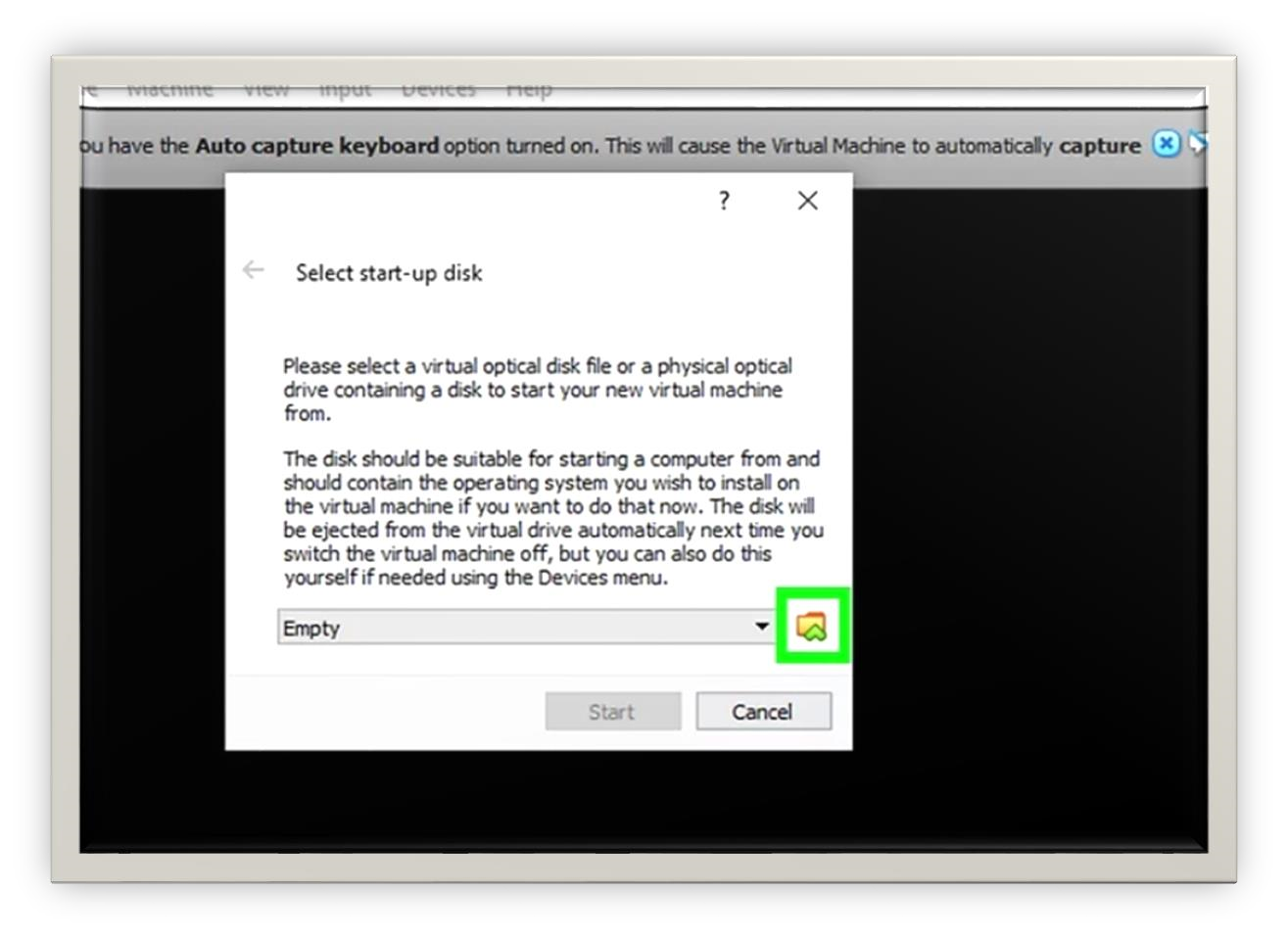
Name: Angadi Saiganesh

Batch:

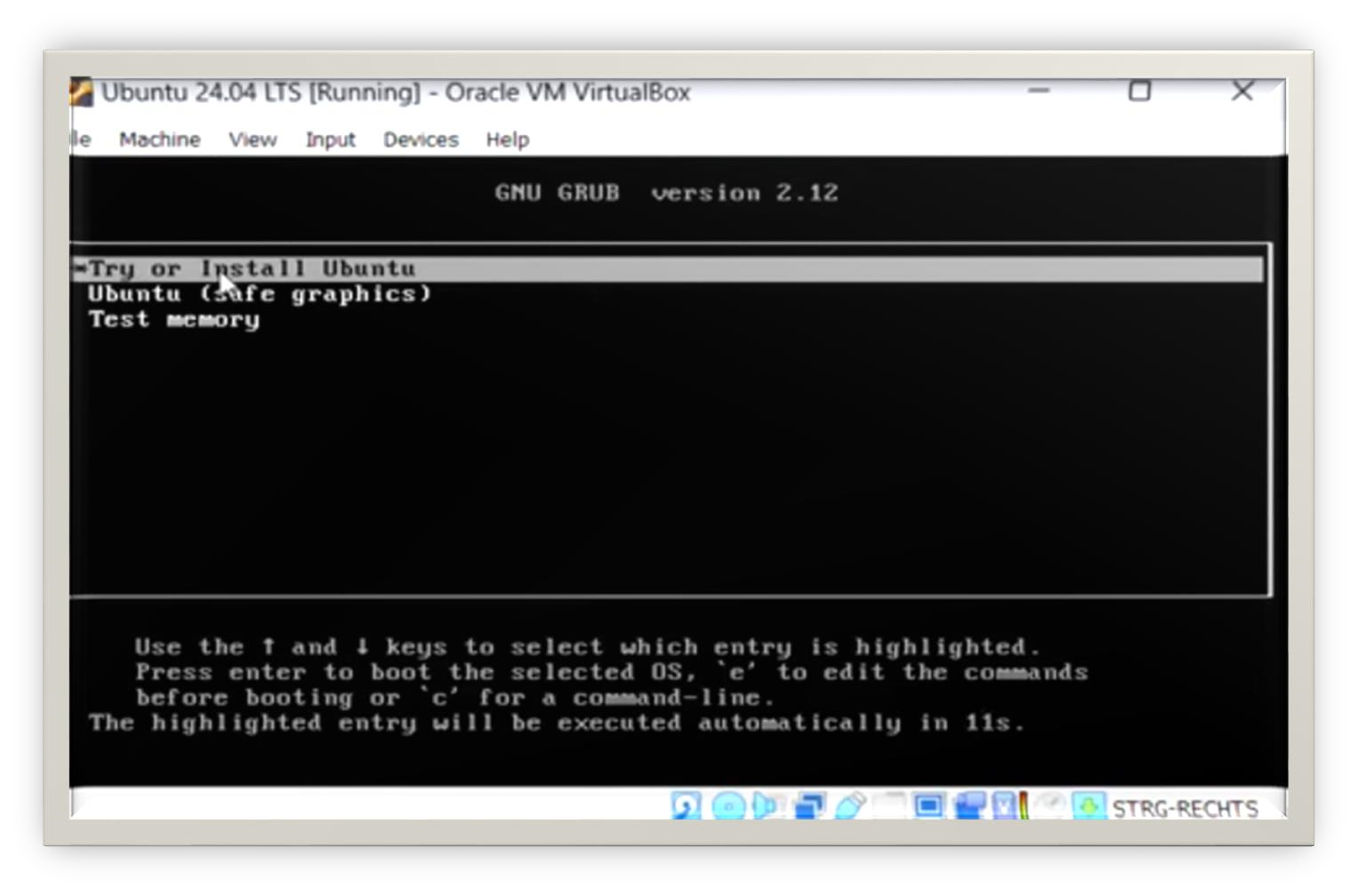
B5



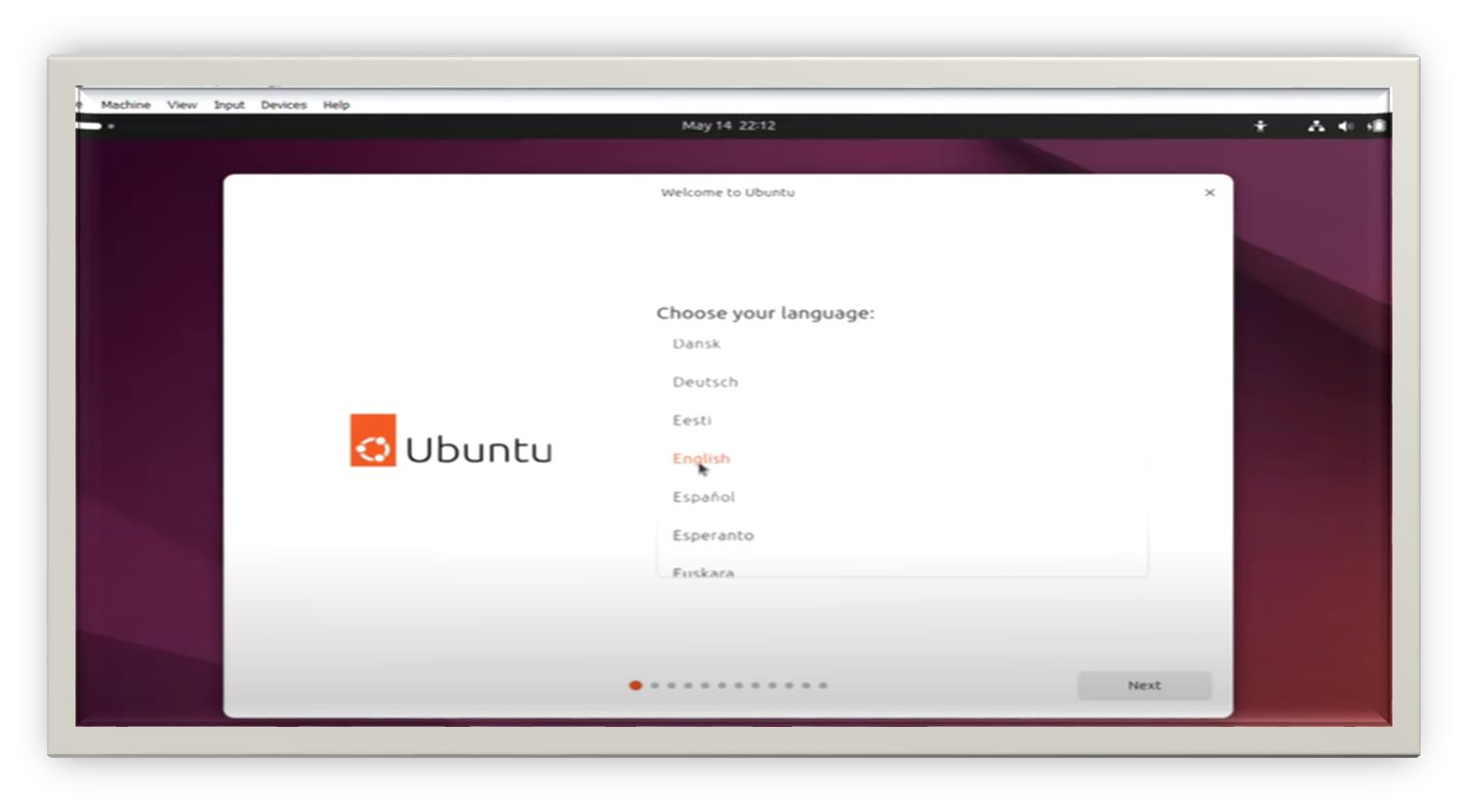
* Click on folder symbol and add path to your downloaded ubuntu iso



Name: Angadi Saiganesh Batch: B5

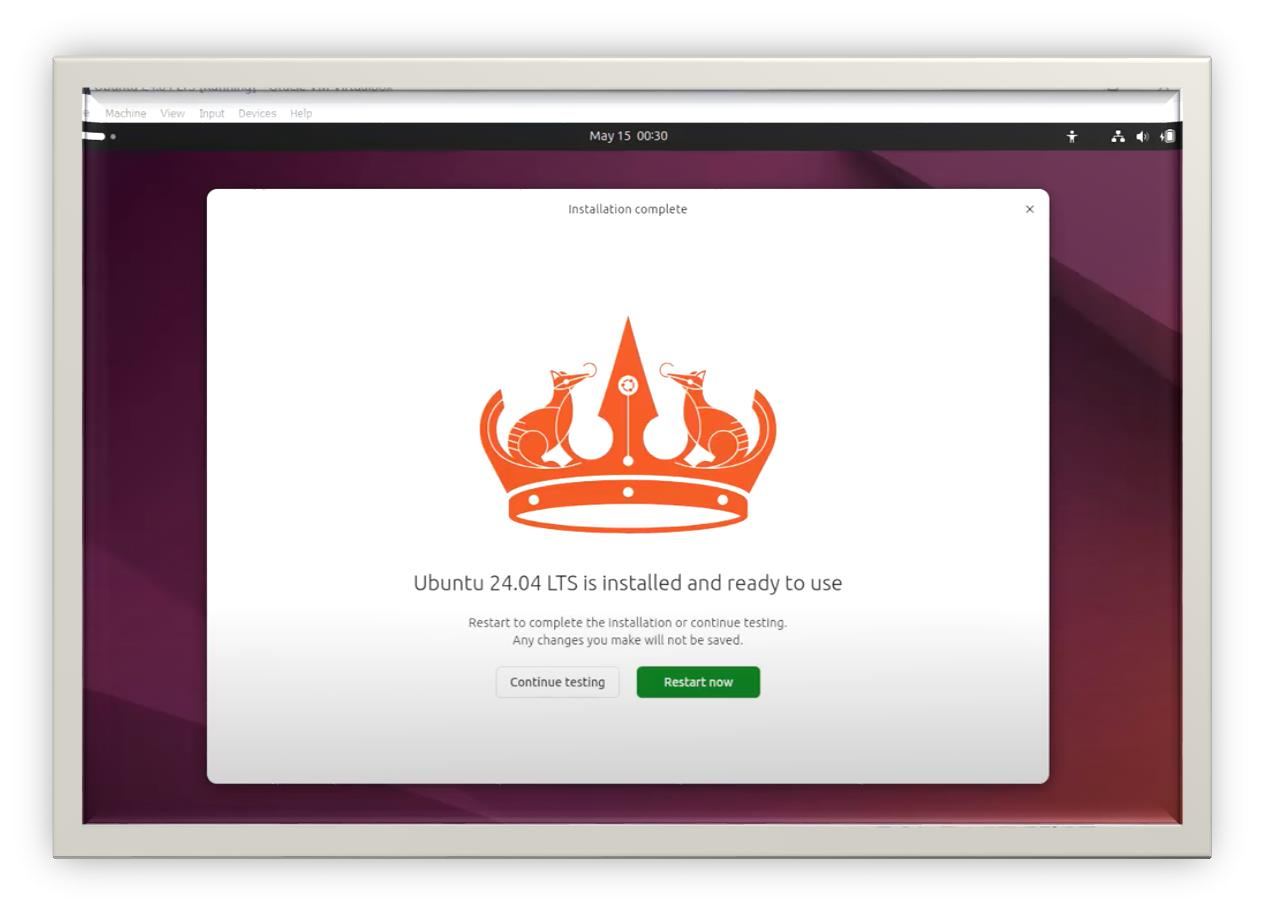
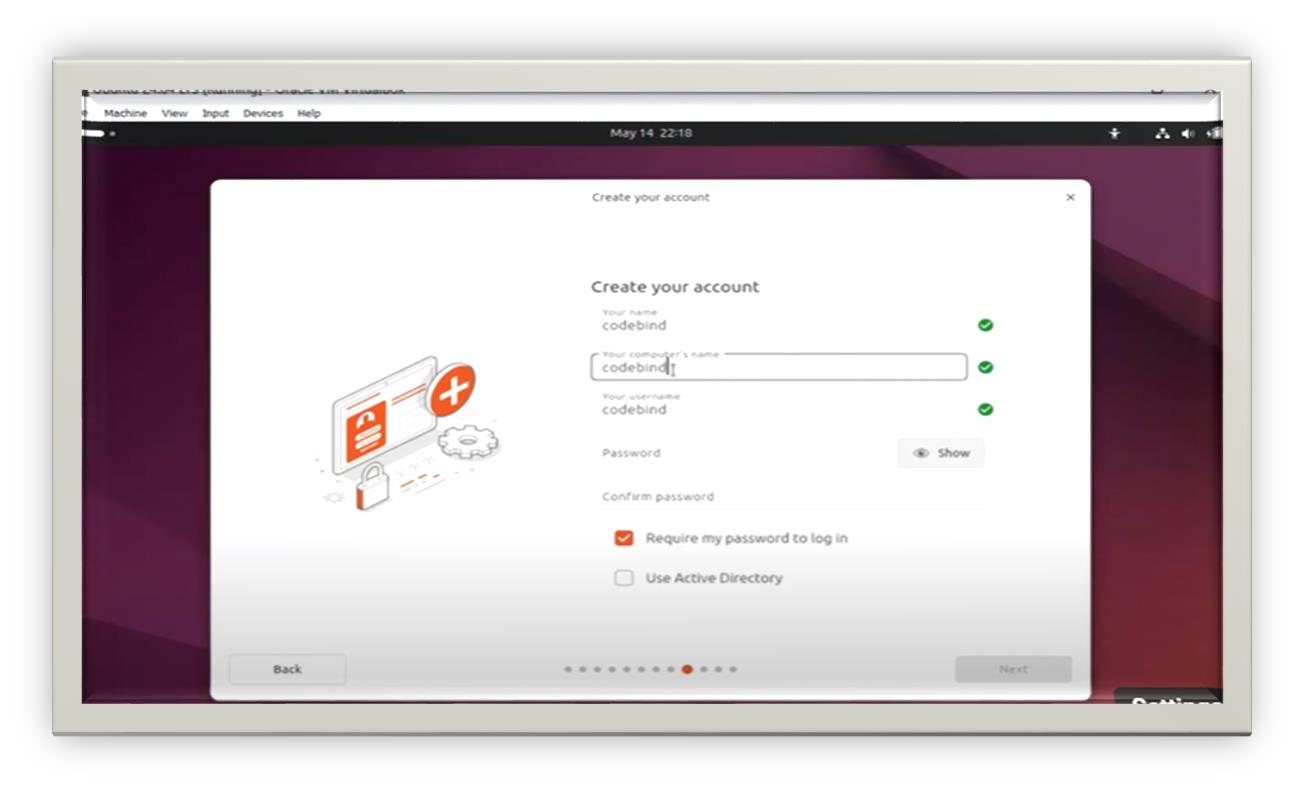


* Click on install
* Select install ubuntu icon
* Configure as per your requirements and do set the account and select rest of the packages



•

Click on Restart/Reboot



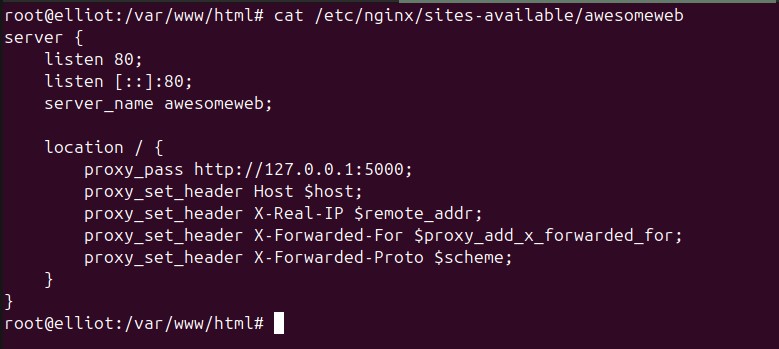
**Important commands**

sudo apt-get update -y sudo apt-get upgrade -y sudo apt install python3-full sudo apt install python3-pip python3 -m venv myenv source myenv/bin/activate pip install requests pip install flask sudo apt-get install nginx sudo systemctl enable nginx sudo systemctl nginx -t sudo systemctl start nginx sudo systemctl stop nginx sudo ln -s /etc/nginx/sites-available/awesomeweb /etc/nginx/sites-enabled sudo systemctl restart nginx

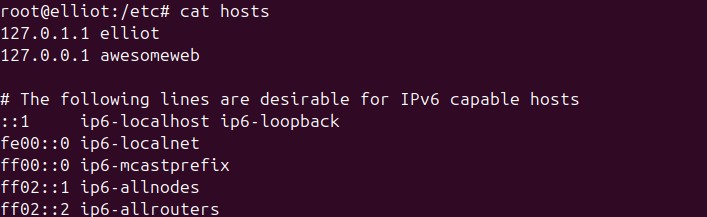
**Task Q1**

Q1. Deploy a website on localhost using either apache2 or Nginx. Create a DNS name for this website as ‘awesomeweb’. You can use any web template you want or can write your own simple HTML code.

**This is nginx config file**



**This is hosts file and it would be helpful to change domain name of the server/ machine**



# Python Script

**Task Q2**

|  |
| --- |
| from flask import Flask, jsonify, render\_template, request import requests from threading import Thread from time import sleep  app = Flask(\_\_name\_\_)    # List of subdomains to check subdomains = [  'http://www.google.com',  'http://www.github.com',  'http://www.facebook.com'  ] status\_dict = {subdomain: 'Unknown' for subdomain in subdomains}  def check\_status(url): try:  response = requests.get(url, timeout=5) if response.status\_code == 200:  return 'Up' else:  return 'Down' except requests.RequestException:  return 'Down'  def update\_status(): while True:  for subdomain in list(status\_dict.keys()):  status\_dict[subdomain] = check\_status(subdomain) sleep(5)    @app.route('/') def index():  return render\_template('index.html')    @app.route('/status') def status():  return jsonify(status\_dict) |
| @app.route('/add\_subdomain', methods=['POST']) def add\_subdomain():  data = request.get\_json() new\_subdomain = data.get('subdomain') if new\_subdomain and new\_subdomain not in status\_dict:  status\_dict[new\_subdomain] = 'Unknown' return '', 204  if \_\_name\_\_ == "\_\_main\_\_": thread = Thread(target=update\_status) thread.daemon = True thread.start() app.run(debug=True, host='0.0.0.0') |

# Frontend

In here I used tailwind css cdn link to style website

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<script src="../static/index.js" defer></script>

<title>Subdomain Status</title>

<link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css" rel="stylesheet">

</head>

<body class="bg-gray-100">

<div class="container mx-auto p-4">

<h1 class="text-4xl font-bold text-center text-blue-600 mb-8">Subdomain Status Monitor</h1>

<div class="bg-white p-6 rounded-lg shadow-lg mb-8">

<input id="subdomainInput" type="text" placeholder="Enter subdomain URL" class="border p-3 w-full rounded-lg focus:outline-none focus:ring-2 focus:ring-blue-500">

<button id="addSubdomainButton" class="bg-blue-500 text-white p-3 mt-4 w-full rounded-lg hover:bg-blue-700 transition duration-300">Add

|  |
| --- |
| Subdomain</button>  <p id="feedbackMessage" class="text-center mt-4"></p>  </div>  <div class="overflow-x-auto">  <table class="min-w-full bg-white shadow-md rounded-lg">  <thead>  <tr>  <th class="py-4 px-6 bg-gray-200 text-left text-gray-600 fontsemibold">Subdomain</th>  <th class="py-4 px-6 bg-gray-200 text-left text-gray-600 fontsemibold">Status</th> </tr>  </thead>  <tbody id="statusTableBody">  </tbody>  </table>  </div>  </div>  </body>  </html> |

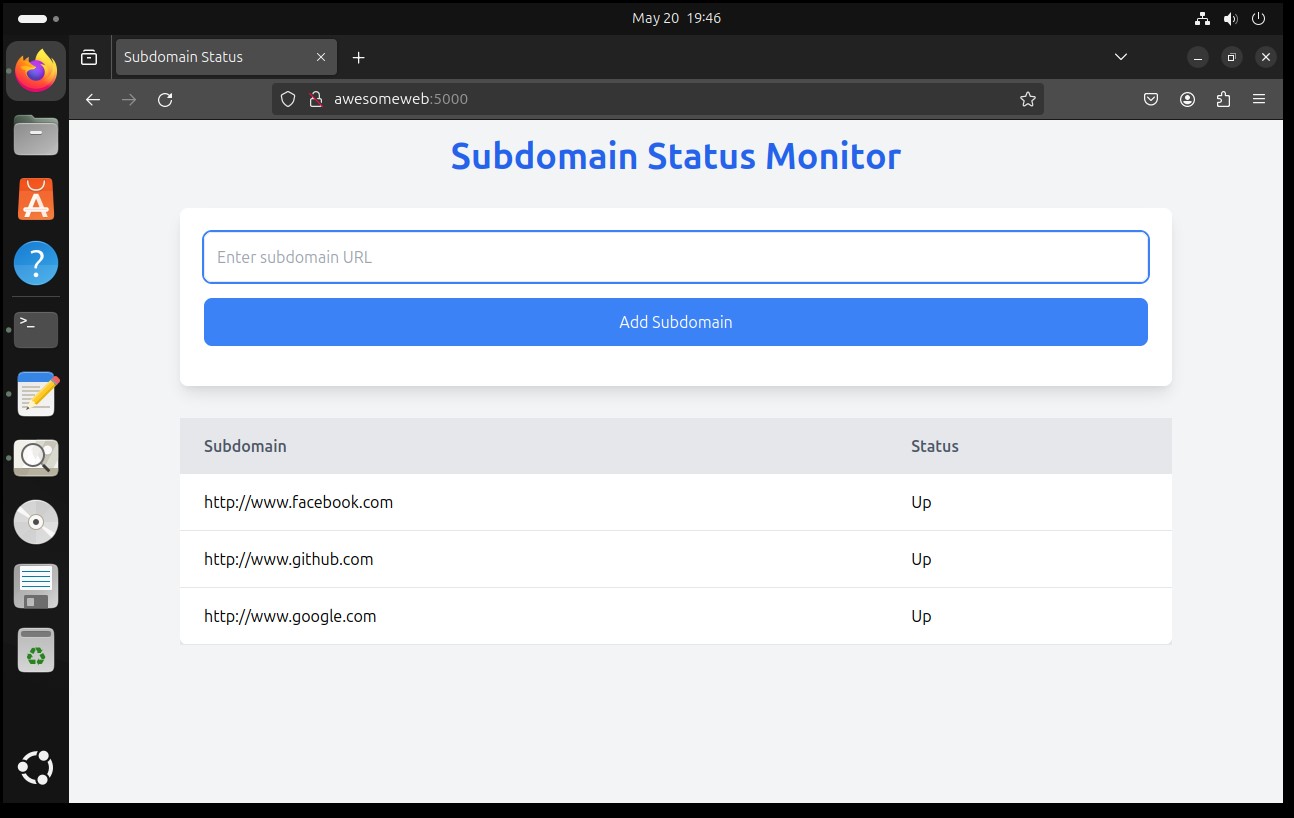
# Javascript

In here I used content stream to update the table using element id

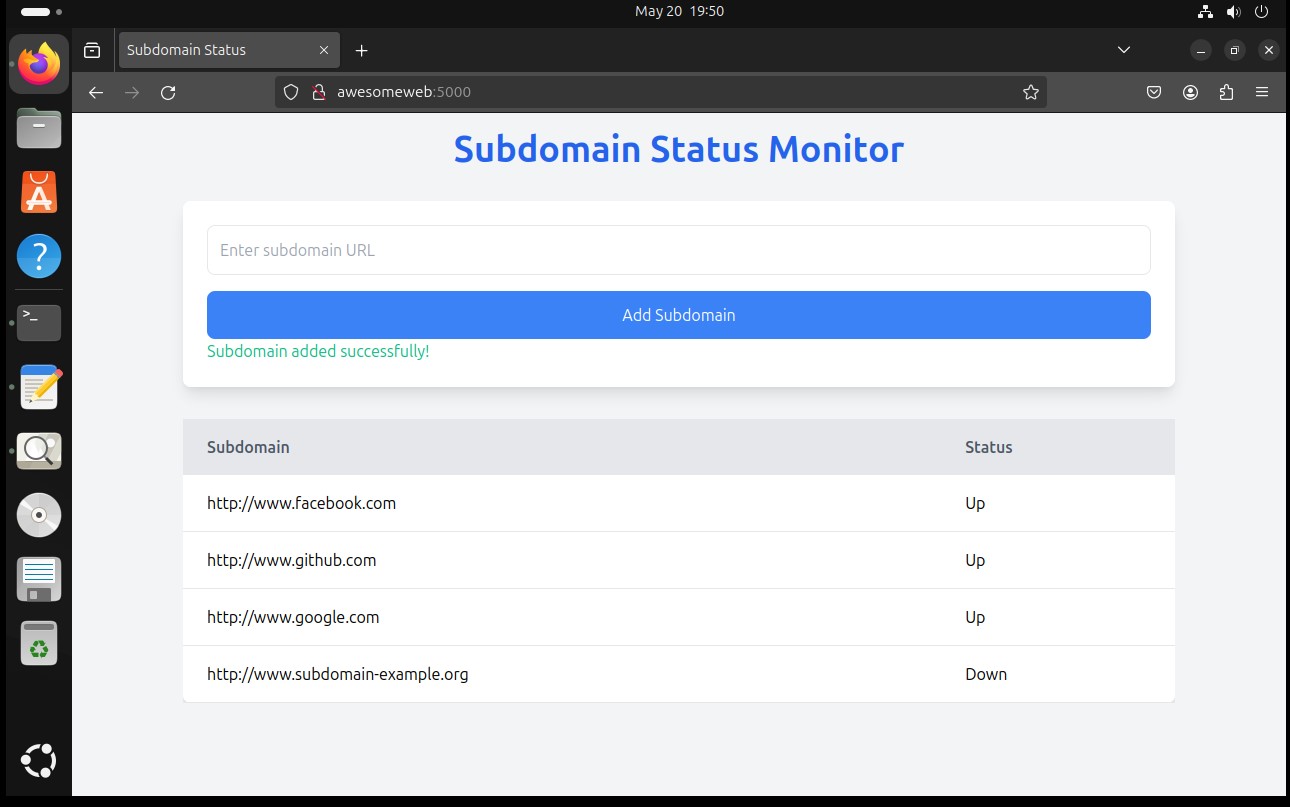
|  |
| --- |
| document.addEventListener("DOMContentLoaded", function() {  const statusTableBody = document.getElementById('statusTableBody'); const subdomainInput = document.getElementById('subdomainInput'); const addSubdomainButton = document.getElementById('addSubdomainButton'); const feedbackMessage = document.getElementById('feedbackMessage');  function fetchStatuses() { fetch('/status')  .then(response => response.json())  .then(data => { statusTableBody.innerHTML = ''; for (const [subdomain, status] of Object.entries(data)) { const row = document.createElement('tr'); row.innerHTML = `  <td class="py-4 px-6 border-b">${subdomain}</td>  <td class="py-4 px-6 border-b">${status}</td> `; |
| statusTableBody.appendChild(row);  }  });  } function addSubdomain() { const subdomain = subdomainInput.value.trim(); if (subdomain) { fetch('/add\_subdomain', { method: 'POST', headers: {  'Content-Type': 'application/json'  }, body: JSON.stringify({ subdomain: subdomain })  }).then(response => { if (response.ok) { subdomainInput.value = ''; feedbackMessage.textContent = 'Subdomain added successfully!'; feedbackMessage.className = 'text-green-500'; fetchStatuses();  } else { feedbackMessage.textContent = 'Failed to add subdomain.'; feedbackMessage.className = 'text-red-500'; }  }).catch(() => { feedbackMessage.textContent = 'Error occurred while adding subdomain.'; feedbackMessage.className = 'text-red-500';  }); } else { feedbackMessage.textContent = 'Please enter a valid subdomain.'; feedbackMessage.className = 'text-red-500';  }  }  addSubdomainButton.addEventListener('click', addSubdomain); setInterval(fetchStatuses, 5000); fetchStatuses();  }); |

# Output

**To check if systems substems are online**



**Example of down server**



**Check with NMAP**

