

MANUALE AMMINISTRATORE

Sommario

Supporto.....	3
Elenco librerie.....	4
Funzionamento script CPU.py.....	4
Funzionamento script temp.py.....	5
Importazione delle Librerie.....	5
Funzione get_cpu_temperature.....	5
Funzione TemperatureCpu.....	5
Funzione get_disk_usage.....	6
Funzione create_plot.....	6
Funzione create_pie_chart.....	6
Funzione display_system_info.....	6
Sezione Principale.....	6
Funzionamento script.py Tempe.py.....	7
1. Importazione delle librerie.....	7
2. Definizione della funzione funzioneTemEpoca().....	7
3. Ottenimento della data e ora attuali.....	7
4. Generazione di valori casuali.....	7
5. Pausa di esecuzione.....	7
6. Stampa dell'intestazione HTTP.....	7
7. Creazione del documento HTML.....	7
Sezioni dell'HTML:.....	8
Contenuto del corpo:.....	8
8. Chiamata della funzione.....	8
Riepilogo.....	8
Funzionamento pagina index.html.....	8
1. Dichiarazione del Documento.....	8
2. Elemento <html>.....	8
3. Elemento <head>.....	8
4. Stili CSS.....	9
5. Elemento <body>.....	9
5.1 Header.....	9
5.2 Container.....	9
5.2.1 Menu.....	9
5.2.2 Content.....	9
5.2.3 Aside.....	9
5.3 Footer.....	9
6. Script JavaScript.....	9
In sintesi.....	10
Run software e link.....	10

Introduzione per l'amministratore

Il monitoraggio delle reti è un aspetto fondamentale per gli amministratori di rete, poiché offre una serie di vantaggi e strumenti essenziali per garantire il buon funzionamento e la sicurezza delle

infrastrutture di rete. Ecco alcune delle principali ragioni per cui il monitoraggio delle reti è importante:

1. **Prestazioni della rete:** Consente di monitorare il traffico di rete, identificare colli di bottiglia e ottimizzare le prestazioni. Gli amministratori possono analizzare il traffico e i tempi di risposta per garantire che le applicazioni e i servizi siano sempre disponibili e reattivi.
2. **Rilevamento dei guasti:** Il monitoraggio continuo aiuta a identificare e diagnosticare rapidamente eventuali problemi, come guasti hardware o malfunzionamenti nei componenti di rete. Ciò consente di ridurre i tempi di inattività e mantenere la continuità del servizio.
3. **Sicurezza:** Permette di rilevare attività sospette o anomalie che potrebbero indicare attacchi informatici, come intrusioni o malware. Gli strumenti di monitoraggio possono inviare avvisi in tempo reale agli amministratori per consentire una risposta tempestiva.
4. **Gestione della capacità:** Gli amministratori possono analizzare i dati di utilizzo della rete per pianificare e gestire le risorse in modo più efficace. Ciò include l'assegnazione della larghezza di banda, l'aggiunta di nuove risorse o l'ottimizzazione delle configurazioni esistenti.
5. **Audit e conformità:** Fornisce registrazioni dettagliate e reportistica sui dati di rete, che possono essere utili per audit di sicurezza e conformità alle normative. Questi report possono dimostrare che le pratiche di gestione della rete sono in linea con le politiche aziendali e le normative vigenti.
6. **Analisi storica:** I dati storici raccolti possono essere analizzati per identificare tendenze e modelli nel traffico di rete. Questo aiuta gli amministratori a prevedere future esigenze e a pianificare aggiornamenti o modifiche necessarie.
7. **Automazione e gestione centralizzata:** Gli strumenti di monitoraggio delle reti consentono una gestione più centralizzata e, in alcuni casi, automatizzata, semplificando le operazioni quotidiane e riducendo il carico di lavoro degli amministratori.

In sintesi, il monitoraggio delle reti è cruciale per garantire che l'infrastruttura di rete funzioni in modo efficiente, sicuro e conforme, contribuendo così al successo complessivo dell'organizzazione.

Sommario

Supporto

Relazione sul Supporto Didattico nel Corso di Python

Durante il percorso di formazione in Python, il supporto del docente è stato fondamentale, specialmente durante le esercitazioni relative al compendio di Python base. Il docente ha fornito una guida costante e ha svolto un ruolo cruciale nell'assicurare che tutti gli studenti avessero le competenze necessarie per affrontare gli argomenti trattati.

1. W3Schools

W3Schools è una piattaforma educativa ben nota che offre risorse su vari linguaggi di programmazione. Nella sezione dedicata a Python e CGI (Common Gateway Interface), ho trovato informazioni utili su come generare HTML dinamico. W3Schools presenta tutorial chiari e facili da seguire, con esempi di codice che illustrano l'uso di Python per creare pagine web interattive.

Questo sito è particolarmente utile per i principianti, poiché offre anche un editor online per testare il codice in tempo reale.

2. TutorialsPoint

TutorialsPoint è un'altra risorsa eccellente per apprendere Python e CGI. Questo sito offre una serie di tutorial che coprono vari aspetti della programmazione con Python, inclusi esempi pratici su come implementare CGI per generare pagine web. Gli articoli sono ben strutturati e forniscono una guida passo-passo che facilita l'apprendimento. Inoltre, TutorialsPoint offre anche quiz e esercizi pratici per mettere alla prova le proprie conoscenze.

3. GeeksforGeeks

GeeksforGeeks è un sito dedicato alla programmazione e alla tecnologia, ricco di articoli, guide e esempi di codice. Ho trovato molte informazioni utili su vari argomenti di programmazione, inclusi script Python per la generazione di pagine web. Gli articoli sono scritti in modo chiaro e forniscono approfondimenti su come utilizzare Python per creare applicazioni web, con focus su best practices.

e suggerimenti pratici.

Esempio di utilizzo di Python CGI

ChatGPT

Per concludere, ho utilizzato ChatGPT per ottenere un esempio pratico di un semplice script Python CGI. Questo esempio illustra come ricevere dati da un modulo HTML e restituire una risposta dinamica. Tali script sono fondamentali per chiunque desideri comprendere come Python possa interagire con il web.

Sommario

Elenco librerie

1. **psutil**
2. **matplotlib**

Sommario

Funzionamento script CPU.py

Il codice Python che hai fornito genera una pagina web per visualizzare la temperatura della CPU in un formato HTML. Vediamo i vari componenti e la loro funzionalità in un unico testo:

1. Importazione di librerie:

- Viene importata la libreria `random` con l'alias `rd`, che serve per generare numeri casuali.

2. Funzione `CPU()`:

- Questa funzione inizia stampando l'intestazione HTTP necessaria per un contenuto HTML.
- Segue la creazione di un documento HTML, che include informazioni sul carattere, sul viewport e il titolo della pagina. Si applicano anche alcuni stili CSS per rendere la pagina visivamente accattivante.
- Gli stili definiscono l'aspetto della pagina, inclusi colori, margini, padding e comportamento responsive per schermi più piccoli.

3. Generazione della temperatura:

- Utilizzando `rd.randint(30, 90)`, viene generata una temperatura casuale per la CPU, compresa tra 30 e 90 gradi Celsius.

- La temperatura viene confrontata con una soglia di 70 gradi per determinare il messaggio da visualizzare: se supera questa soglia, viene mostrato un messaggio in rosso per indicare un potenziale problema, altrimenti viene visualizzato un messaggio in verde con il valore attuale della temperatura.

4. Interazione utente:

- È presente un pulsante che consente di ricaricare la pagina, permettendo all'utente di vedere una nuova lettura della temperatura.

5. Chiusura del documento HTML:

- Il codice conclude con la chiusura dei tag HTML e la chiamata della funzione `CPU()`, che attiva l'intero processo di generazione della pagina.

In sintesi, il codice crea un'interfaccia web semplice che visualizza la temperatura della CPU e fornisce un modo per aggiornare il valore. È progettato per essere utilizzato su un server locale (indicando indirizzi localhost) e può servire come base per ulteriori sviluppi o monitoraggi di sistema.

Sommario

Funzionamento script temp.py

Python che raccoglie e visualizza informazioni sul sistema, in particolare la temperatura della CPU e l'utilizzo del disco. Di seguito spiego ciascuna parte del codice in dettaglio:

Importazione delle Librerie

Viene eseguita l'importazione di diverse librerie.

- **datetime:** per gestire le date e gli orari.
- **matplotlib.pyplot:** per creare grafici.
- **base64:** per codificare le immagini in formato base64.
- **psutil:** per raccogliere informazioni sulle risorse di sistema, come la temperatura della CPU e l'utilizzo del disco.
- **io:** per gestire i buffer di input/output.
- **time:** per gestire le pause nel codice, utile per il campionamento dei dati.

Funzione `get_cpu_temperature`

Questa funzione tenta di recuperare la temperatura corrente della CPU. Usa la libreria `psutil` per ottenere le informazioni sulle temperature e restituisce la temperatura del primo core (se disponibile). Se si verifica un errore, restituisce `None`.

Funzione `TemperatureCpu`

Questa funzione raccoglie i dati sulla temperatura della CPU per un intervallo di tempo specificato. Esegue un ciclo dieci volte, in cui:

1. Ottiene l'ora attuale e la converte in timestamp.

2. Chiama la funzione `get_cpu_temperature` per ottenere la temperatura.
3. Se la temperatura è valida (non `None`), la aggiunge a un elenco `cpu_data` insieme al timestamp.
4. Introduce una pausa di 0,5 secondi tra le letture per evitare letture troppo ravvicinate.

Funzione `get_disk_usage`

Questa funzione restituisce l'utilizzo totale, usato e libero del disco principale. Utilizza la libreria `psutil` per ottenere queste informazioni.

Funzione `create_plot`

Questa funzione crea un grafico utilizzando Matplotlib. Riceve i dati da visualizzare, il titolo del grafico e le etichette degli assi.

1. Crea un buffer di byte in memoria.
2. Configura il grafico (dimensione, stile, etichette, titolo).
3. Salva il grafico nel buffer in formato PNG.
4. Converte l'immagine in una stringa base64 e la restituisce.

Funzione `create_pie_chart`

Simile alla funzione `create_plot`, questa funzione crea un grafico a torta per visualizzare i dati dell'utilizzo del disco. Si occupa di configurare il grafico e restituire l'immagine in formato base64.

Funzione `display_system_info`

Questa è la funzione principale che genera e visualizza le informazioni di sistema in formato HTML.

1. Imposta l'intestazione del contenuto e inizia a costruire la struttura HTML.
2. Raccoglie i dati sulla temperatura della CPU tramite `TemperatureCpu` e genera il grafico corrispondente.
3. Raccoglie informazioni sull'utilizzo del disco e crea un grafico a torta.
4. Crea una tabella per visualizzare i dati di utilizzo del disco in gigabyte.
5. Mostra i dati di temperatura e umidità generati casualmente in un'altra sezione con il rispettivo grafico e tabella.
6. Chiude la struttura HTML.

Sezione Principale

L'ultima parte del codice verifica se il file è eseguito come script principale e, in tal caso, chiama la funzione `display_system_info` per avviare il processo di monitoraggio e visualizzazione delle informazioni di sistema.

In sintesi, il codice crea un'applicazione web che monitora e visualizza in tempo reale la temperatura della CPU e l'utilizzo del disco, fornendo anche un'interfaccia visiva attraverso grafici e tabelle.

Sommario

Funzionamento script.py Tempe.py

1. Importazione delle librerie

- **import datetime:** Questa libreria fornisce classi per gestire date e orari. Viene utilizzata per ottenere l'ora corrente e per calcolare il timestamp.
- **import random:** Permette di generare numeri casuali. Qui viene usata per generare valori casuali per temperatura e umidità.
- **import time:** Questa libreria fornisce funzioni per gestire il tempo. Viene utilizzata per mettere in pausa l'esecuzione del programma per un secondo.

2. Definizione della funzione **funzioneTemEpoca()**

- **def funzioneTemEpoca():** Qui si definisce una funzione chiamata `funzioneTemEpoca`. Tutto il codice indentato sotto di essa farà parte di questa funzione e sarà eseguito quando la funzione verrà chiamata.

3. Ottenimento della data e ora attuali

- **now = datetime.datetime.now():** Viene creato un oggetto `now` che contiene la data e l'ora correnti.
- **ts = str(datetime.datetime.timestamp(now)):** Si calcola il timestamp dell'oggetto `now`, che rappresenta il numero di secondi trascorsi dal 1 gennaio 1970 (Epoca Unix), e lo si converte in una stringa.

4. Generazione di valori casuali

- **temperatura = str(random.randint(20, 100)):** Si genera un valore casuale per la temperatura, compreso tra 20 e 100, e lo si converte in stringa.
- **umidita = str(random.randint(20, 100)):** Analogamente, si genera un valore casuale per l'umidità, anch'esso tra 20 e 100, e lo si converte in stringa.

5. Pausa di esecuzione

- **time.sleep(1):** Il programma si ferma per 1 secondo. Questo può essere utile per simulare un aggiornamento in tempo reale o per ridurre il carico di elaborazione.

6. Stampa dell'intestazione HTTP

- **print("Content-type:text/html; charset=utf-8\r\n\r\n"):** Questa riga invia l'intestazione HTTP al client, specificando che il contenuto che seguirà è di tipo HTML e codificato in UTF-8.

7. Creazione del documento HTML

- Il blocco **print(f"..." . . . " ")** genera un documento HTML che verrà visualizzato nel browser. La sintassi f-string permette di inserire variabili direttamente nel testo HTML.

Sezioni dell'HTML:

- **<html>**, **<head>**, **<title>**: Questi tag definiscono la struttura del documento e il titolo che apparirà nella scheda del browser.
- **<style>**: Qui vengono definiti gli stili CSS per formattare l'aspetto della pagina. Ci sono stili per il corpo della pagina, il titolo, i pulsanti e anche per creare un effetto di lampeggiamento per il testo.

Contenuto del corpo:

- **<body>**: In questa sezione vengono visualizzati i dati:
 - **Temperatura**: Un titolo che mostra la temperatura attuale con un effetto di lampeggiamento.
 - **Epoca**: Un paragrafo che mostra il timestamp calcolato.
 - **Umidità**: Un altro paragrafo che mostra il valore dell'umidità.
- **Pulsanti**: Due pulsanti sono inclusi:
 - Il primo riporta l'utente alla homepage.
 - Il secondo ricarica la pagina corrente.

8. Chiamata della funzione

- **funzioneTemEpoca()**: Questa riga finale chiama la funzione definita, eseguendo così tutto il codice all'interno di essa e mostrando il risultato nel browser.

Riepilogo

In sintesi, questo codice genera e visualizza una pagina web che mostra valori casuali per temperatura e umidità, insieme al timestamp attuale. La pagina è stilizzata e include funzionalità interattive tramite pulsanti.

Sommario

Funzionamento pagina index.html

1. Dichiarazione del Documento

- **<!DOCTYPE html>**: Indica che il documento è un file HTML5.

2. Elemento <html>

- **<html lang="it">**: Inizia l'elemento HTML e specifica che il linguaggio della pagina è italiano.

3. Elemento <head>

- Qui si trovano i metadati e le risorse esterne per la pagina.
- **<meta charset="UTF-8">**: Imposta la codifica dei caratteri su UTF-8, che supporta la maggior parte dei caratteri internazionali.

- **<meta name="viewport" content="width=device-width, initial-scale=1.0">**: Imposta il viewport per il responsive design, ottimizzando la visualizzazione sui dispositivi mobili.
- **<script>**: Includono librerie JavaScript esterne:
 - **jsPDF**: Per generare documenti PDF.
 - **QRious**: Per generare codici QR.
 - **html2canvas**: Per catturare screenshot di elementi HTML.

4. Stili CSS

- **<style>**: Definisce lo stile della pagina.
- Viene impostato uno sfondo e vengono specificati colori, padding, margini, e stili per diversi elementi (header, menu, contenuto, footer, ecc.).
- Utilizza proprietà CSS come `flex` per creare layout responsivi, `rgba` per colori con trasparenza e `box-shadow` per effetti di ombra.

5. Elemento <body>

- Contiene il contenuto principale della pagina.

5.1 Header

- **<header class="header">**: Definisce l'intestazione della pagina con un titolo e un pulsante per scaricare un PDF.
- Il pulsante ha un evento `onclick` che attiva la funzione `downloadPDF()`.

5.2 Container

- **<div class="container">**: Un contenitore flessibile per organizzare il contenuto.

5.2.1 Menu

- **<div class="menu">**: Sezione per il menu di navigazione con collegamenti a documenti e script.

5.2.2 Content

- **<div class="content">**: Area centrale della pagina con un titolo e una descrizione, oltre a un contenitore per un codice QR e un'immagine.

5.2.3 Aside

- **<div class="aside">**: Una colonna laterale per collegamenti utili.

5.3 Footer

- **<footer class="footer">**: Sezione finale della pagina che include un messaggio di benvenuto.

6. Script JavaScript

- **<script>**: Contiene funzioni JavaScript per gestire la generazione del PDF e dei codici QR.

- **downloadPDF()**: Crea un documento PDF con informazioni sul software e collegamenti utili.
- **generateQRCode()**: Genera un codice QR con un URL specificato.
- **window.onload = generateQRCode;**: Inizializza la generazione del codice QR quando la pagina è caricata.

In sintesi

Questo codice definisce una pagina web per il monitoraggio delle temperature della CPU, con un layout responsivo, un menu di navigazione, opzioni per scaricare un PDF e visualizzare un codice QR. Utilizza librerie JavaScript per funzionalità avanzate, come la generazione di PDF e codici QR, e include uno stile CSS per rendere la pagina esteticamente gradevole e funzionale.

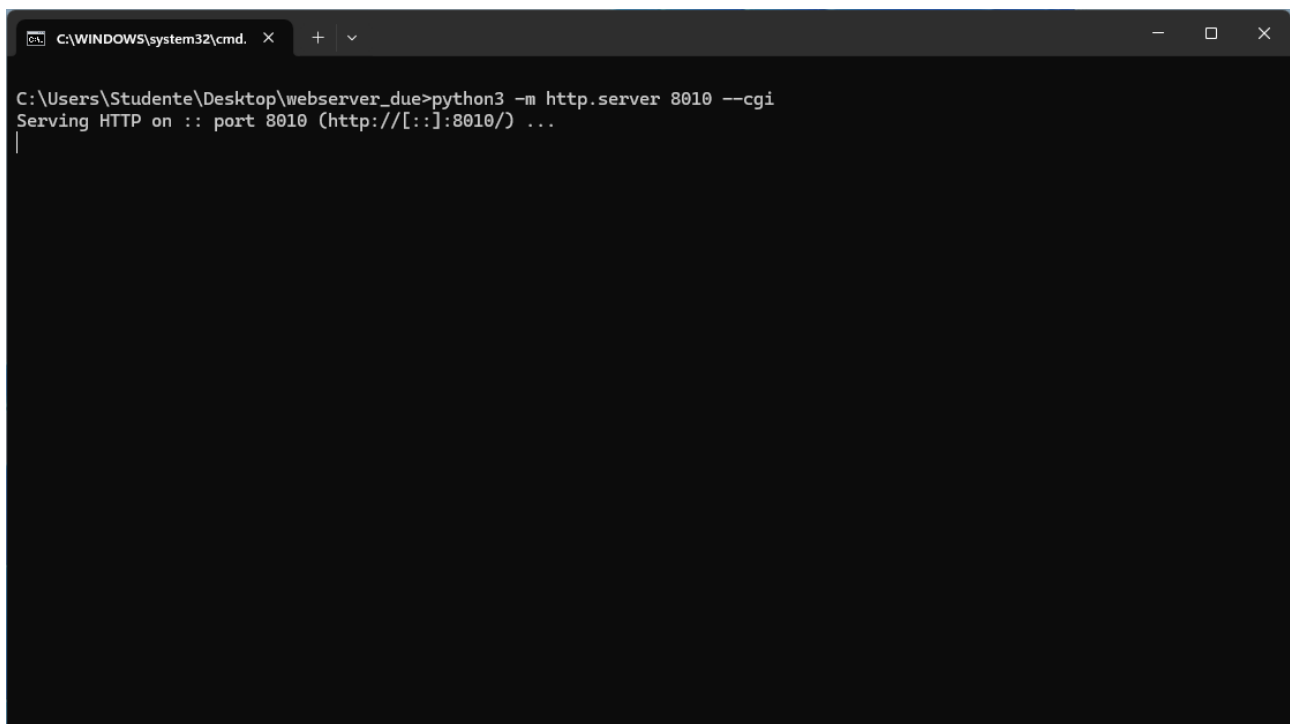
Sommario

Run software e link

<http://localhost:8010/cgi-bin/temp.py>

<http://localhost:8010/cgi-bin/Tempe.py>

<http://localhost:8010/cgi-bin/CPU.py>



```
C:\WINDOWS\system32\cmd. X + v
C:\Users\Studente\Desktop\webserver_due>python3 -m http.server 8010 --cgi
Serving HTTP on :: port 8010 (http://[::]:8010/) ...
```