Progetto Programmazione di Reti 2021

Traccia 2: Python Web Server

Si immagini di dover realizzare un Web Server in Python per una azienda ospedaliera. I requisiti del Web Server sono i seguenti:

- Il web server deve consentire l'accesso a più utenti in contemporanea
- La pagina iniziale deve consentire di visualizzare la lista dei servizi erogati dall'azienda ospedaliera e per ogni servizio avere un link di riferimento ad una pagina dedicata.
- L'interruzione da tastiera (o da console) dell'esecuzione del web server deve essere opportunamente gestita in modo da liberare la risorsa socket.
- Nella pagina principale dovrà anche essere presente un link per il download di un file pdf da parte del browser
- Come requisito facoltativo si chiede di autenticare gli utenti nella fase iniziale della connessione.

Alessandro Zanzi
Mail istituzionale: alessandro.zanzi2@studio.unibo.it
Matricola: 0000915472

Green Hospital

Introduzione:

Il progetto ha l'obbiettivo di creare un Web Server in Python, con possibilità di accessi multipli in parallelo da parte di utenti diversi.

Descrizione:

Per avviare il server usare il comando ”python3server.py”, specificare una porta come argomento, se non viene specificata quella di default sarà la 8080.
Una volta avviato il server sarà possibile interrogare il sito presente su localhost.
Le connessioni sono garantite grazie al metodo:
socketserver.ThreadingTCPserver(("",port), http.server.SimpleHTTPRequestHandler), questo metodo consente di realizzare una connessione TCP ad un HTTP server multi-thread, in quanto controlla e abilita la generazione di thread figli per la gestione di più accessi da parte degli utenti in parallelo.
Se l'esecuzione dello script python avviene da console, il parametro port può essere inserito da tastiera, altrimenti è definita con il valore 8080.
L'host prestabilito per la connessione è l'host da cui vieni eseguito il codice, perciò tale valore è impostato come 'localhost' oppure " ".
Le funzionalità che il client richiede al server http sono eseguite grazie al comando http.server.SimpleHTTPRequestHandler, fra cui la ricezione del messaggio, la decodifica, la ricerca del file da inviare, la codifica di quest'ultimo ed il suo invio.
La terminazione attraverso l'interrupt da tastiera è gestita per mezzo della funzione close_signal_handler che attraverso la serie Ctrl+c chiude il server.
La funzione main() contiene:
- abilitazione del server a ricevere connessioni multiple grazie alla funzione
 serverHTTP.daemon_threads = True;
- riutilizzo delle connessioni con la funzione serverHTTP.allow_reuse_address = True;
- gestione di interrupt da tastiera, richiamando la funzione
close_signal_handler: signal.signal(signal.SIGINT,close_signal_handler);
- infine pone il server in stato di attesa, entrando in un loop infinito, di richieste HTTP con il metodo serverHTTP.serve_forever()

Utilizzo:

Per eseguire il codice è possibile lanciarlo da terminale oppure eseguirlo tramite l'applicazione Spyder.
Una volta eseguito il codice, occorrerà cliccare su questo link http://localhost:8080 oppure aprire un browser/user_agent e copiare la stringa http://localhost:8080 e verrà aperta una pagina HTML con la quale poter interagire come spiegato qui di seguito.

HTML:

Il server, come prima pagina, apre il file index.html che mostra i servizi principali del Green Hospital, quali:
- Contatti
- Locazione
- Orari
- Visite ed esami disponibili
- Ambulatori Principali
- Medici

Per ognuna di queste voci è presente un collegamento ad altre pagine html contenute nella directory file_html.
Infine è possibile cliccare sul pulsante rosso “Informazioni_generali.pdf” in fondo alla pagina principale per effettuare il download di un file pdf contente le informazioni principali riguardo il Green Hospital.

Librerie utilizzate:

- http
- socketserver
- signal

Link utili:

Codice Sorgente: https://github.com/AleZanzi /Progetto-Reti-2021