Relazione Progetto per Programmazione di Reti

Filippo Greppi
Matricola: 0001114837
filippo.greppi2@studio.unibo.it

24 giugno 2025

Consegna del progetto

Traccia 1 – Web Server + Sito Web Statico (livello base/intermedio)
Titolo: Realizzazione di un Web Server minimale in Python e pubblicazione di un sito statico

Obiettivo:
Progettare un semplice server HTTP in Python (usando socket) e servire un sito web statico con HTML/CSS.

Requisiti minimi:

- Il server deve rispondere su localhost:8080.
- Deve servire almeno 3 pagine HTML statiche.
- Gestione di richieste GET e risposta con codice 200.
- Implementare risposta 404 per file inesistenti.

Estensioni opzionali:

- Gestione dei MIME types (.html, .css, .jpg, ecc.).
- Logging delle richieste.
- Aggiunta di animazioni o layout responsive.

Output atteso:

- Codice del server Python.
- Cartella www/ con i file HTML/CSS.
- Relazione tecnica.

Requisiti per l'esecuzione

Per eseguire il progetto sono necessari:

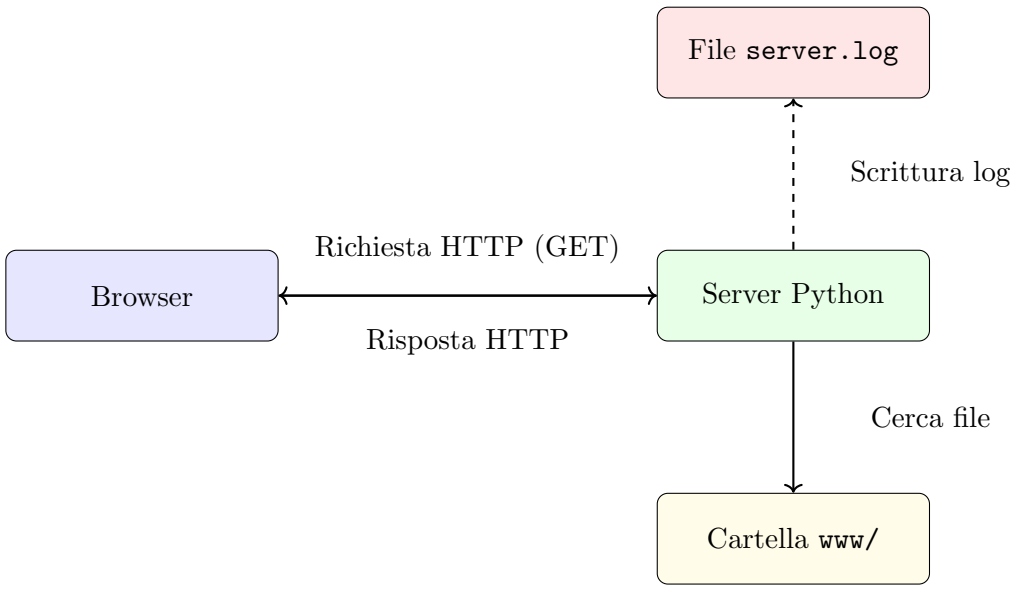
- Python 3 installato sul sistema.
- Sistema operativo compatibile (Windows, Linux, macOS).
- La cartella `www/` contenente almeno tre file HTML statici.
- Il file `server.py` nella stessa directory della cartella `www/`.
- (Facoltativo) Un browser web per testare il server.

Descrizione del funzionamento

Il progetto consiste in un server HTTP minimale scritto in Python che permette di pubblicare un sito web statico. Il server utilizza la libreria `socket` per gestire le connessioni di rete e risponde alle richieste HTTP provenienti dal browser.

- **Avvio del server:** Il server si avvia in ascolto su `localhost` alla porta 8080. Rimane in attesa di richieste da parte dei client (tipicamente un browser).
- **Gestione delle richieste:** Il server accetta solo richieste HTTP di tipo GET. Se viene ricevuto un altro tipo di richiesta, risponde con un errore 405 (Method Not Allowed).
- **Risposta 400 Bad Request:** Se la richiesta HTTP è malformata (ad esempio, la request line non ha il formato corretto), il server restituisce una pagina di errore 400.
- **Servizio delle pagine statiche:** Quando riceve una richiesta GET, il server cerca il file corrispondente nella cartella `www/`. Se il path richiesto è `/`, viene restituito il file `index.html` di default.
- **Risposta 200 OK:** Se il file esiste, il server lo invia al client con codice di stato 200 e il corretto tipo MIME (ad esempio `text/html` per le pagine HTML, `text/css` per i fogli di stile, ecc.).
- **Risposta 404 Not Found:** Se il file richiesto non esiste nella cartella `www/`, il server restituisce una pagina di errore 404.
- **Gestione degli errori:** In caso di errori interni, il server restituisce una risposta 500 (Internal Server Error).
- **Logging:** Ogni richiesta viene registrata sia su console che su file (`server.log`), indicando data, ora, metodo, percorso richiesto e codice di risposta.
- **Gestione dei MIME types:** Il server determina automaticamente il tipo di contenuto da restituire in base all'estensione del file richiesto, garantendo la corretta visualizzazione di HTML, CSS, immagini, ecc.
- **Estensioni grafiche:** Le pagine HTML/CSS nella cartella `www/` possono includere animazioni o layout responsive per migliorare l'aspetto grafico del sito.

Schema del funzionamento del server



Legenda del diagramma

1. Il browser invia una richiesta HTTP al server (solo metodo GET supportato).
2. Il server cerca il file nella cartella `www/`.
3. Il server restituisce il file richiesto o una risposta di errore:
   200 OK, 404 Not Found, 405 Method Not Allowed, 400 Bad Request, 500 Internal Server Error.
4. Ogni richiesta viene registrata nel file `server.log`.

Istruzioni per l'uso

1. Assicurarsi di avere Python 3 installato.
2. Posizionare il file `server.py` e la cartella `www/` (con almeno 3 file HTML) nella stessa directory.
3. Aprire il terminale nella cartella del progetto.
4. Avviare il server con il comando:

   ```
   python3 server.py
   ```

5. Aprire il browser e visitare `http://localhost:8080`

6. Navigare tra le pagine statiche disponibili (ad esempio `/about.html`, `/contact.html`).

7. Per fermare il server, premere `Ctrl+C` nel terminale.

Nota: Tutte le richieste vengono salvate nel file `server.log`. Per aggiungere nuove pagine basta inserirle nella cartella `www/`.