0.0 Introduzione

0.1 - Stack dei protocolli di rete: Modello ISO/OSI

Il modello ISO/OSI è composto da sette livelli: Applicazione, Presentazione, Sessione, Trasporto, Rete, Collegamento Dati e Fisico. Quando parliamo di Web, ciascun livello corrisponde a specifiche tecnologie: l'HTML si colloca nel livello Applicazione, UNICODE/UTF-8 nel livello Presentazione, HTTP/HTTPS nel livello Sessione, TCP nel livello Trasporto, IP nel livello Rete, protocolli come CSMA/CD, 802.11, LTE nel livello Collegamento Dati, mentre il livello Fisico rappresenta l'hardware di rete.

0.2 - Sistemi Web

I sistemi Web si basano su protocolli aperti che consentono di realizzare un'infrastruttura di comunicazione accessibile a tutti. Questa caratteristica permette a chiunque di entrare in rete, pubblicare un servizio Web o accedere ai servizi esistenti senza restrizioni particolari.

Tipi di Siti Web

I siti Web si distinguono principalmente in due categorie: statici e dinamici. I siti Web statici consistono in un insieme di pagine HTML con contenuto fissato staticamente dal Web Master. Sono particolarmente utili per realizzare "Siti Vetrina", ma risultano inadatti per implementare servizi interattivi per l'utente. I siti Web dinamici, invece, sono costituiti da pagine HTML il cui contenuto viene generato dinamicamente da programmi che operano sul Web Server. In questo caso, la pagina viene interamente rigenerata ad ogni richiesta di elaborazione.

0.3 - Architetture

Le architetture per realizzare siti web sono evolute nel tempo attraverso tre principali proposte.

La 1-tier Architecture è la più semplice ed è adatta per i siti statici. In questa configurazione, il tier è rappresentato dal protocollo HTTP. Il browser invia una richiesta HTTP per ottenere una risorsa, il server riceve la richiesta e, se la risorsa è presente sul disco, la invia al client impacchettandola nella risposta HTTP.

La 2-tier Architecture è più adatta per i siti dinamici, dove i contenuti devono essere generati da programmi che lavorano sul server. In questo caso abbiamo due tier: il primo è il protocollo HTTP, mentre il secondo è il protocollo CGI (Common Gateway Interface) che gestisce la comunicazione tra il web server e l'applicativo CGI. Il browser invia una richiesta HTTP al server, il quale la riceve e, riconoscendo che è destinata a un programma, attiva l'applicativo CGI passandogli il contenuto della richiesta. L'applicativo genera quindi la pagina HTML o altro contenuto, che il server impacchetta nella risposta HTTP e invia al browser.

La 3-tier Architecture introduce l'interazione con un DBMS. Il terzo tier è rappresentato dalla connessione con il database, che avviene generalmente attraverso il protocollo ODBC (Open DataBase Connectivity). In questo modello, l'applicativo CGI, dopo aver ricevuto la richiesta dal server, apre una connessione con il database, invia le query necessarie, riceve le tabelle risultanti, genera l'output e lo invia al web server, che a sua volta lo impacchetta nella risposta HTTP e lo trasmette al browser.

0.4 - Richiesta/Risposta HTTP

I messaggi HTTP fungono da "corrieri" per il trasporto delle informazioni e sono suddivisi in due parti: Header (intestazione) e Body (corpo). L'Header specifica varie informazioni come l'URL della risorsa richiesta, dettagli sul richiedente (agent) e il tipo del contenuto del body. Il Body contiene i contenuti effettivi da inviare. Nella 1-tier Architecture, il body della richiesta è sempre vuoto, mentre il body della risposta contiene la risorsa richiesta.

Metodi HTTP

Il protocollo HTTP prevede diversi metodi per l'invio dei dati, tra cui i più comuni sono GET e POST. Con il metodo GET, i dati vengono aggiunti all'URL della richiesta: un simbolo "?" dopo il percorso apre la "query string", che è composta da triplette campo=valore separate dal simbolo "&". Ad esempio, in un URL come https://www.google.it/search?q=unibg, stiamo effettuando una query a Google con il testo "unibg". I caratteri speciali nell'URL, come gli spazi, vengono codificati (ad esempio, %20 rappresenta lo spazio).

Il metodo POST viene utilizzato quando non si vuole che i dati siano visibili nella barra degli indirizzi o quando il valore da trasmettere è troppo grande (come nel caso di aree di testo o file). In questo caso, il corpo della richiesta HTTP contiene i campi del form, mentre l'URL del destinatario rimane invariato.

URL e MIME Type

L'URL (Uniform Resource Locator) serve per localizzare qualsiasi risorsa sul Web con una struttura uniforme: protocollo://dominio/percorso. Il protocollo specifica il tipo di comunicazione (ad esempio HTTP o HTTPS), il dominio può essere un indirizzo IP e può includere la porta TCP, mentre il percorso indica la posizione della risorsa nel file system virtuale del sito.

Il MIME Type specifica il formato del contenuto del Body della risposta HTTP. Esempi comuni includono text/plain per testo semplice, text/html per pagine web, image/png e img/jpeg per immagini, audio/mpeg per file audio, application/pdf per documenti PDF e application/msword per documenti Word.

0.5 - Risorse Computazionali e Cloud Computing

Per realizzare un sistema Web sono necessarie diverse risorse computazionali: storage (un file system), un DBMS e i database, un server con un Web Server per gestire le chiamate HTTP. Gestire server in casa comporta numerosi problemi: rischi di rottura o furto, surriscaldamento, costi energetici elevati, necessità di personale dedicato, obsolescenza delle macchine, costoso collegamento a Internet e scarsa scalabilità.

Inizialmente, la soluzione è stata l'hosting, con provider che forniscono risorse computazionali sulle proprie macchine, facendo hosting del sito, fornendo un collegamento veloce a Internet e accollandosi i costi di obsolescenza e aggiornamento del software di base.

L'evoluzione tecnologica ha portato alle macchine virtuali, programmi che consentono di creare sistemi di elaborazione completi ma virtuali. Su una stessa macchina fisica possono essere emulate molteplici macchine virtuali, con il vantaggio di sfruttare meglio le risorse disponibili. Nella soluzione tradizionale, la ripartizione delle risorse è fissa, mentre nel mondo cloud viene riconfigurata dinamicamente in base alle necessità, secondo un modello Pay per Use in cui si paga in base all'effettivo utilizzo.

Il Cloud Computing rappresenta quindi il passaggio da una soluzione di hosting, dove le macchine virtuali sono ospitate sempre sulla stessa macchina fisica, a una soluzione dove l'infrastruttura ridistribuisce le macchine virtuali su una "Nuvola di Server".

0.6 - Dettagli del Corso

Il corso tratterà numerosi argomenti relativi alla programmazione web: HTML e CSS per la struttura e lo stile delle pagine, JavaScript come linguaggio di programmazione client-side, XML come formato di

rappresentazione dei dati, tecniche di manipolazione delle pagine web da JavaScript, AJAX per la comunicazione asincrona con i server, JSON come nuovo formato di rappresentazione dei dati, jQuery come framework per potenziare JavaScript.

Verranno inoltre affrontati i Server-Side Script come Servlet e PHP, Node.js che porta JavaScript sul lato server, MongoDB e i database NoSQL, l'accesso ai database nei server-side script, Python come linguaggio per la Data Science e Django come framework Python per sistemi web.

Le esercitazioni pratiche, tenute dall'Ing. Paolo Fosci, avranno l'obiettivo di far fare pratica con l'insieme delle tecnologie presentate, in vista dello sviluppo degli elaborati o progetti. L'esame consisterà in una prova scritta teorica a crocette (16 punti) e due progetti da sviluppare (8+8 punti).