# World Wide Web

**Scritto da Febogamedev**

Il termine inglese web significa “ragnatela” e richiama l’idea di un insieme molto esteso di risorse collegate tra loro mediante collegamenti ipertestuali, chiamati link.
Ogni link rimanda a un indirizzo web, normalmente espresso mediante un **URL**, che identifica una risorsa e indica come raggiungerla attraverso Internet (Berners-Lee et al., 2005).

Le risorse possono comprendere testi, immagini, video, documenti, dati, servizi e applicazioni interattive.

- Il Web utilizza Internet come infrastruttura di comunicazione: Internet è la rete globale che permette a dispositivi e applicazioni di scambiarsi dati. Il Web è uno dei principali sistemi costruiti sopra Internet e permette di identificare, collegare, richiedere e utilizzare risorse (Berners-Lee, 1990; Fielding et al., 2022).
- Il Web è un sistema distribuito: le sue risorse non sono conservate tutte su un unico computer, ma sono gestite da numerosi server situati in luoghi differenti (Berners-Lee, 1990).

**Una pagina web** è un documento o un’interfaccia che il browser costruisce elaborando una o più risorse. Può contenere collegamenti ad altre pagine, a differenti sezioni della stessa pagina o ad altri tipi di risorse.

Le tre tecnologie fondamentali utilizzate per realizzare le pagine web sono:

- **HTML**, che descrive i contenuti, la struttura e il significato degli elementi. Una pagina web è generalmente basata su un documento *HTML*, che costituisce il documento principale elaborato dal browser.
- **CSS**, che controlla l’aspetto, il layout e l’adattamento della pagina ai diversi dispositivi tramite delle regole che definiscono lo stile della pagina.
- **JavaScript**, che può aggiungere comportamenti, elaborazioni e funzioni interattive.

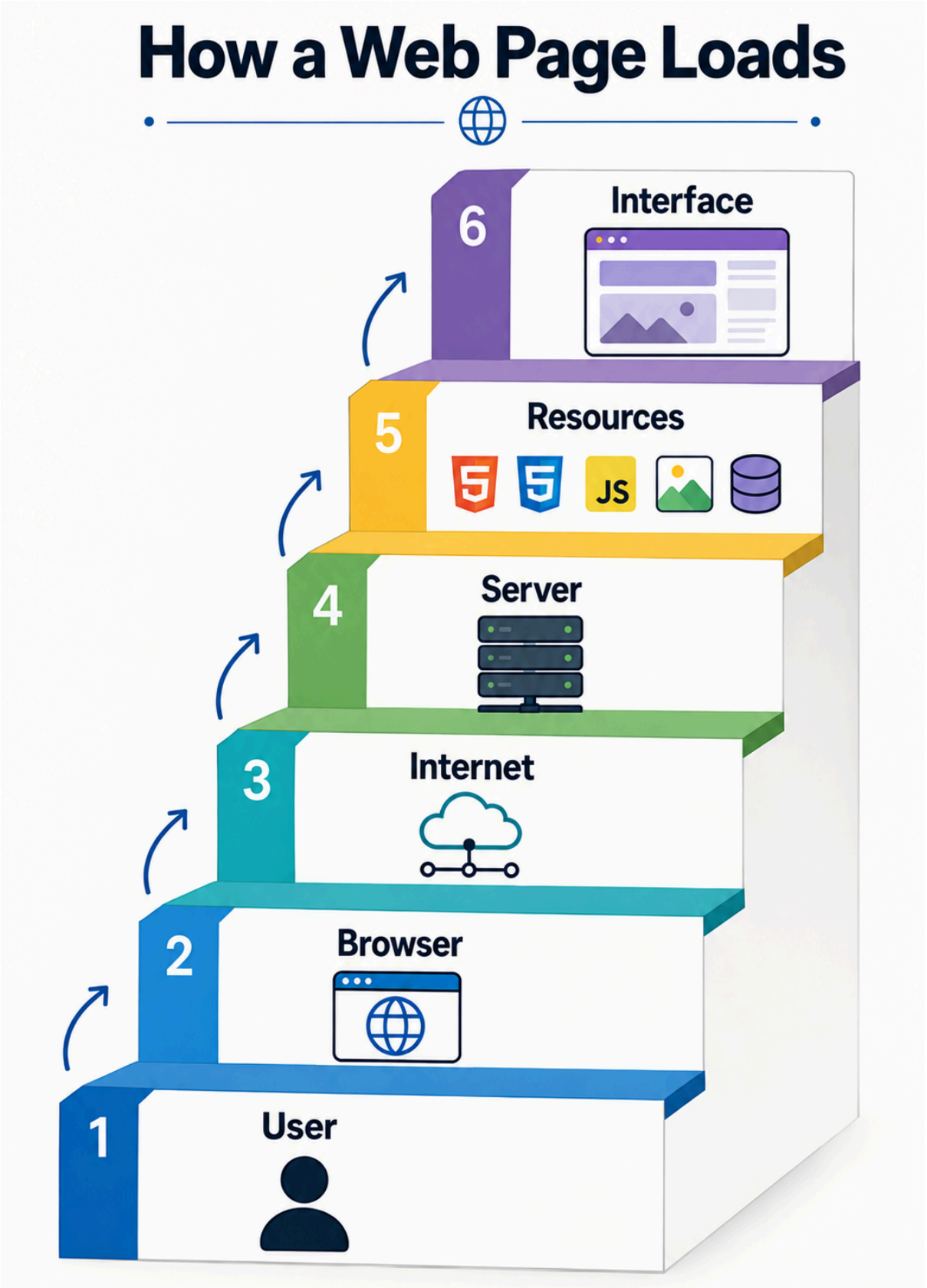
*L’HTML* costituisce la struttura di base della pagina. CSS e JavaScript non sono indispensabili per l’esistenza di una pagina web, ma permettono rispettivamente di controllarne la presentazione e di aggiungere comportamenti più complessi (CSS Working Group, 2026; Ecma International, 2026; WHATWG, 2026).

**Un sito web** è invece un insieme organizzato di pagine e risorse correlate, generalmente riconducibili allo stesso progetto, servizio o organizzazione e raggiungibili attraverso uno o più indirizzi Web.

Per accedere alle risorse web, l’utente utilizza normalmente un software client chiamato **browser**. Il browser interpreta l’indirizzo richiesto, comunica con i server, riceve i dati e li elabora per costruire l’interfaccia con cui interagisce l’utente (WHATWG, 2026).

La comunicazione tra browser e web server avviene principalmente attraverso **HTTP**, un protocollo applicativo basato sullo scambio di richieste e risposte.

Oggi questa comunicazione è normalmente protetta attraverso **HTTPS** (Fielding et al., 2022).



# Hypertext

**Un ipertesto** è una struttura informativa in cui un documento contiene collegamenti verso altre risorse oppure verso parti differenti dello stesso documento (Berners-Lee, 1990).

A differenza di un testo progettato per essere letto seguendo un ordine lineare, un ipertesto permette una lettura non necessariamente sequenziale.
Il lettore può scegliere quali collegamenti seguire e costruire così il proprio percorso tra le informazioni.

Il meccanismo che collega un elemento del documento a una destinazione si chiama **collegamento ipertestuale**, in inglese **hyperlink** o, più semplicemente, **link** (Berners-Lee, 1990).

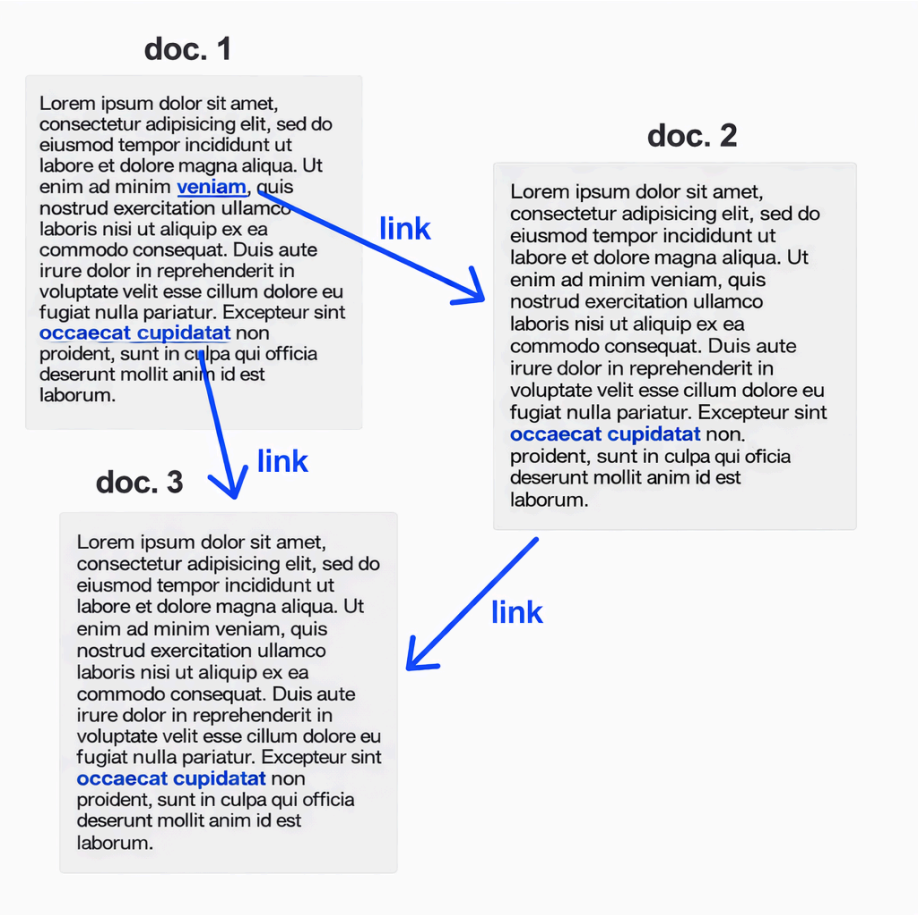
Un collegamento ipertestuale mette in relazione due elementi:
- Un'origine, cioè l'elemento selezionabile dal quale parte il collegamento.
- Una destinazione, cioè la risorsa o la sezione indicata dal collegamento.

La destinazione può essere:
- Un'altra pagina dello stesso sito.
- Una pagina appartenente a un altro sito.
- Una sezione della stessa pagina o di un'altra pagina.
- Un'immagine, un video, un documento o un altro tipo di risorsa.

L'origine del collegamento non deve essere necessariamente una singola parola.
Un link può essere associato a una frase, un'immagine, un'icona, una voce di menu o un'altra componente dell'interfaccia.

Quando i collegamenti mettono in relazione non soltanto testi, ma anche immagini, video, audio e contenuti interattivi, si parla di **ipermedia** (Berners-Lee, 1990).
Il Web può quindi essere considerato non soltanto un sistema ipertestuale, ma anche un **sistema ipermediale**.

Il Web è stato concepito come un grande sistema di risorse collegate.
Un link può condurre a una risorsa ospitata sullo stesso server oppure su un server situato in un'altra parte della rete (Berners-Lee, 1990).
Infatti, la navigazione web è il processo attraverso il quale l'utente si sposta tra pagine, sezioni e risorse utilizzando collegamenti, menu e altri sistemi di orientamento.



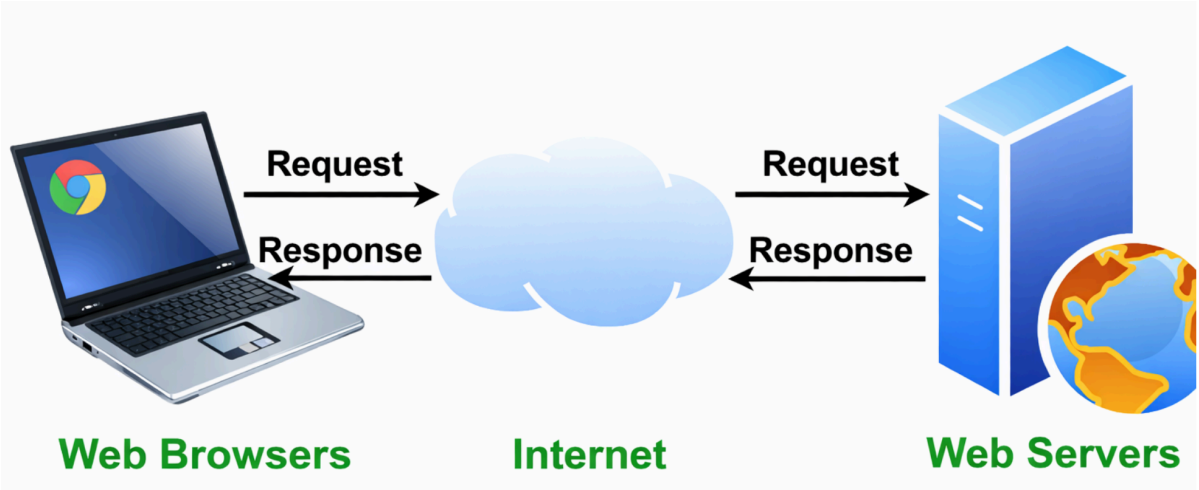

*Esempio di ipertesto*

# Web server

**Un web serve**r è un software che riceve richieste *HTTP* da parte dei client, come i browser, e restituisce le relative risposte (Fielding et al., 2022).

Nel linguaggio comune, l'espressione web server può indicare anche il computer o l'infrastruttura sulla quale questo software viene eseguito. In questo capitolo useremo però il termine principalmente per indicare il software che gestisce la comunicazione web.

Quando un browser richiede una pagina, un'immagine o un altro contenuto, invia una richiesta al web server. Il server interpreta la richiesta, individua o genera i dati associati alla risorsa richiesta e restituisce una risposta che il browser o un'altra applicazione possono interpretare (Fielding et al., 2022).




Dal punto di vista del server, il contenuto restituito può essere statico o dinamico.

**Risorsa statica**
Una risorsa statica viene inviata dal server nella forma in cui è già stata preparata e conservata.
Sono frequentemente risorse statiche:

- Immagini.
- File audio e video.
- Font.
- Fogli di stile CSS.
- File JavaScript.
- Documenti HTML preparati precedentemente.

Il termine statica indica che il server non deve generare appositamente il contenuto per ogni richiesta, ma può inviarlo nella forma in cui è stato conservato.

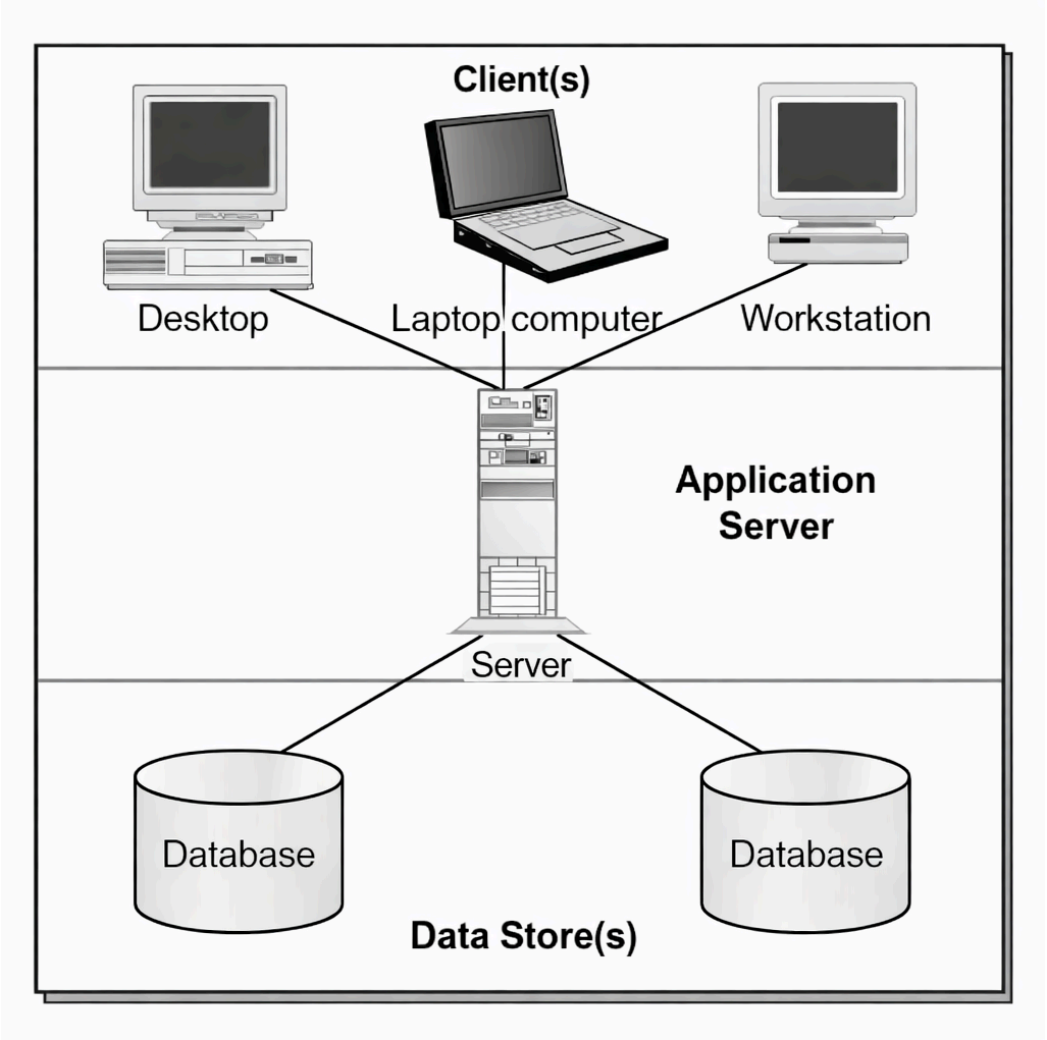
Si noti che una risorsa statica può comunque contribuire alla costruzione di una pagina interattiva.
Per esempio, un file JavaScript può essere statico dal punto di vista del server, anche se produce comportamenti dinamici quando viene eseguito nel browser (Ecma International, 2026; WHATWG, 2026).

**Risorsa dinamica**
Una risorsa dinamica è un contenuto generato o assemblato dal server in risposta a una richiesta.

Il contenuto restituito può dipendere, per esempio, dalla pagina richiesta, dall'utente autenticato, dai dati inseriti in un modulo, dalla lingua selezionata, dai prodotti disponibili, dalle informazioni conservate in un database, dall'orario o da altre condizioni.

Per generare la risposta, il web server può collaborare con un'applicazione lato server, che esegue la logica necessaria.

L’applicazione può comunicare con un database o con altri servizi, recuperare i dati richiesti e inserirli in un documento *HTML* prima di inviarlo al browser.




***Esempio***
Consideriamo un e-commerce (un sito specializzato nel commercio elettronico):

- Quando il browser richiede il file *logo.png*, il server recupera l’immagine già presente nel proprio spazio di archiviazione e la invia senza modificarla. Il logo è quindi una risorsa statica.
- Quando invece il browser richiede la pagina del catalogo, un’applicazione lato server può interrogare un database, recuperare i prodotti disponibili, i prezzi e le immagini e assemblare una risposta specifica. In questo caso, la pagina del catalogo è una risorsa dinamica, perché viene generata sul server in risposta alla richiesta.

Il software web server deve essere eseguito su un computer fisico oppure su un ambiente virtuale (un sistema creato mediante software che si comporta come un computer indipendente).
Uno degli ambienti che può ospitarlo è la ***VPS***, acronimo di Virtual Private Server, cioè server privato virtuale.

Una VPS è una macchina virtuale creata all'interno di un server fisico mediante un sistema di virtualizzazione.
Lo stesso computer fisico può ospitare più *VPS*, ma ciascuna funziona come un ambiente separato, con un proprio sistema operativo e determinate risorse, come CPU, memoria e spazio di archiviazione (Scarfone et al., 2011).

All'interno di una VPS è possibile installare un software web server, insieme all'applicazione, al database e agli altri programmi necessari al funzionamento del sito.
La VPS rappresenta quindi l'ambiente virtuale nel quale vengono eseguiti i programmi, mentre il web server è il software che riceve le richieste *HTTP* e restituisce le risposte (Fielding et al., 2022; Scarfone et al., 2011).

***Esempio***
Un'azienda può acquistare una VPS, installarvi il sistema operativo Linux e configurare Nginx come web server. Quando un utente visita il sito, la richiesta raggiunge la VPS e viene gestita da Nginx, che restituisce direttamente una risorsa statica oppure inoltra la richiesta a un'applicazione lato server.

# Applicazione web

**Un'applicazione web**, o **web app**, è un software al quale l'utente accede attraverso un browser.

A differenza delle applicazioni native o desktop, che normalmente devono essere scaricate e installate sul dispositivo, una web app può essere utilizzata raggiungendo un indirizzo Web (anche se alcune web app possono anche essere installate sul dispositivo, ma continuano a utilizzare le tecnologie e l'ambiente di esecuzione del browser) (WHATWG, 2026; W3C Web Applications Working Group, 2026).

Quando l'utente apre una web app, il browser scarica dal server le risorse necessarie, come documenti HTML, fogli di stile CSS, immagini, codice JavaScript e altri dati.
Il browser interpreta queste risorse ed esegue l'applicazione nel proprio ambiente (WHATWG, 2026).

L'elaborazione necessaria al funzionamento dell'applicazione può avvenire nel browser, sul server oppure essere distribuita tra entrambi. Possiamo quindi distinguere tra:
- Applicazioni prevalentemente lato client.
- Applicazioni prevalentemente lato server.
- Applicazioni ibride.

**Applicazione web prevalentemente lato client**

In un’applicazione web prevalentemente lato client, gran parte della logica e dell’elaborazione viene eseguita dal browser sul dispositivo dell’utente (Ecma International, 2026; WHATWG, 2026).

Il server fornisce inizialmente i file dell’applicazione, come codice JavaScript, immagini e altre risorse.
Dopo il caricamento di queste risorse, molte operazioni possono essere eseguite localmente utilizzando la *CPU* e, quando possibile, la *GPU* del dispositivo.

Il server può comunque essere coinvolto in operazioni come l’autenticazione, la sincronizzazione, il salvataggio online, la collaborazione o l’accesso a servizi esterni.

Vantaggi principali:
- Interazione rapida, perché molte operazioni non richiedono un continuo scambio di dati con il server.
- Minore carico computazionale sul server.
- Possibilità di elaborare dati e file direttamente sul dispositivo.
- Alcune funzioni possono essere utilizzate senza connessione, se l’applicazione è stata progettata per funzionare offline e le risorse necessarie sono state conservate localmente (W3C Web Applications Working Group, 2026).

Svantaggi principali:
- Le prestazioni dipendono dalla potenza del dispositivo.
- Il browser limita l’accesso diretto al sistema operativo e ai file.
- Il caricamento iniziale può richiedere il trasferimento di una grande quantità di codice e risorse.
- Il codice eseguito nel browser viene inviato al dispositivo dell’utente e non può contenere informazioni segrete o essere considerato affidabile per autorizzare operazioni importanti (OWASP Foundation, 2021).

***Esempio***
Piskel è una web app per la creazione di pixel art, sprite e animazioni che viene eseguita prevalentemente nel browser.

Quando l’utente apre Piskel, il browser scarica i file HTML, CSS e JavaScript necessari al funzionamento dell’applicazione. Operazioni come disegnare i pixel, gestire i fotogrammi, visualizzare l’anteprima dell’animazione e generare file GIF o spritesheet vengono elaborate principalmente sul dispositivo dell’utente.
Il server fornisce inizialmente i file dell’applicazione, ma non deve essere contattato ogni volta che l’utente disegna un pixel o modifica un fotogramma.

**Applicazione web prevalentemente lato server**
In un’applicazione web prevalentemente lato server, una parte importante della logica, della gestione dei dati e dell’elaborazione viene eseguita dal server.
Il browser continua a gestire la visualizzazione dell’interfaccia e a raccogliere le azioni dell’utente, mentre il server può:

- Verificare l’identità e i permessi dell’utente.
- Recuperare informazioni da un database.
- Eseguire calcoli o elaborazioni.
- Generare un documento HTML o altri dati.
- Registrare le operazioni effettuate.
- Inviare il risultato al browser (Fielding et al., 2022).

Vantaggi principali:
- I dati e la logica più importanti rimangono sotto il controllo del gestore.
- L’applicazione dipende meno dalla potenza del dispositivo dell’utente.
- La logica lato server può essere aggiornata centralmente.
- È più semplice condividere e sincronizzare dati tra utenti e dispositivi.

Svantaggi principali:
- Molte funzioni dipendono dalla connessione.
- Il server deve sostenere il carico computazionale degli utenti.
- I tempi di risposta possono aumentare quando la rete è lenta.
- La gestione dell’infrastruttura può avere costi elevati.

***Esempio***
Quando un utente accede al proprio conto bancario, il browser mostra l’interfaccia e permette di impartire comandi.
Il server verifica le credenziali e i permessi, recupera il saldo, registra le operazioni e autorizza eventuali trasferimenti.
Queste operazioni devono essere verificate dal server perché il browser viene eseguito su un dispositivo controllato dall’utente e non può essere considerato una fonte affidabile.
Il server mantiene quindi lo stato ufficiale del conto e decide quali operazioni possono essere effettuate.

**Applicazioni web ibride**
Nella pratica, molte applicazioni web moderne sono ibride.

Il browser può gestire l’interfaccia, le animazioni, la validazione immediata dei dati e parte dell’elaborazione.
Il server può invece occuparsi dell’autenticazione, dei database, della sincronizzazione, delle operazioni riservate e dei servizi che richiedono risorse computazionali elevate.

La distinzione tra lato client e lato server non indica quindi categorie completamente separate, ma descrive dove vengono eseguite le diverse parti dell’applicazione.

***Esempio***
Photopea è una web app per la modifica delle immagini eseguita prevalentemente nel browser.

Quando l'utente apre Photopea, il browser scarica il codice e le risorse necessarie. Operazioni come l'uso dei pennelli, la gestione dei livelli, le selezioni, i filtri e la cronologia delle modifiche vengono elaborate principalmente dal dispositivo dell'utente.

- I file locali possono essere aperti, elaborati e salvati senza essere trasferiti ai server di Photopea.
- L'applicazione può comunque comunicare con servizi esterni quando l'utente sceglie di utilizzare sistemi di archiviazione cloud.

# URL

Una risorsa web può essere identificata attraverso un indirizzo, generalmente un ***URL***.

URL è l'acronimo di Uniform Resource Locator. Un URL è una stringa standardizzata che identifica una risorsa e fornisce le informazioni necessarie per riferirsi a essa (Berners-Lee et al., 2005).

Gli URL permettono di raggiungere pagine web, immagini, video, documenti, file, servizi e altre risorse disponibili attraverso il Web. URL differenti possono anche condurre alla stessa risorsa.

Per rappresentare in forma semplificata un URL assoluto che utilizza HTTP o HTTPS, possiamo usare la seguente struttura. Gli elementi tra parentesi quadre sono opzionali (Berners-Lee et al., 2005):

**schema://host[:porta][/percorso][?query][#frammento]**

***Esempio***
https://example.com/catalogo/scarpe?colore=nero&taglia=42#modelli

Le sue componenti sono:

- Schema: https.
- Host: example.com.
- Percorso: /catalogo/scarpe.
- Query: colore=nero&taglia=42.
- Frammento: modelli.

**Schema**
Lo schema è la prima parte *dell'URL* e termina con il carattere ":".

Il suo compito è indicare il tipo di *URL* e le regole secondo le quali deve essere interpretato. Negli *URL* con schema http e https, lo schema indica anche il sistema di comunicazione da utilizzare (Berners-Lee et al., 2005; Fielding et al., 2022).

Lo schema viene normalmente scritto in minuscolo.
Un URL assoluto deve comprendere lo schema. L'utente può scrivere soltanto, per esempio, example.com.

In questo caso, l'utente non sta inserendo formalmente un URL completo.
La barra degli indirizzi interpreta il testo e può costruire automaticamente un URL, generalmente provando a utilizzare HTTPS.
Il comportamento preciso dipende però dal browser e dalle sue impostazioni.

Negli URL con schema http e https, il carattere ":" conclude lo schema, mentre "//" introduce **l'autorità**, la parte che identifica il soggetto che gestisce o ospita la risorsa, che nella forma semplificata comprende l'host e l'eventuale porta:

Non tutti gli schemi utilizzano una parte di autorità. Per esempio:

mailto:nome@example.com
In questo caso mailto è lo schema e non è seguito da "//". Il browser può utilizzare l'indirizzo per aprire il programma di posta elettronica configurato (IANA, n.d.-b).

Alcuni schemi comuni sono (IANA, n.d.-b):

| **Nome** | **Utilizzo** |
|---|---|
| *http* | Accesso a risorse mediante HTTP |
| *https* | Accesso a risorse mediante HTTP su una connessione protetta tramite TLS |
| *ftp* | Trasferimento di file mediante FTP; oggi non è generalmente gestito direttamente dai browser |
| *about* | Accesso a pagine o informazioni interne del browser |
| *file* | Indica che la risorsa si trova nel file system locale del dispositivo. Il browser può quindi essere utilizzato per aprire e visualizzare alcuni file locali, come documenti, immagini o pagine HTML. |
| *mailto* | Apertura del programma di posta per scrivere a un indirizzo |

**Host**
L'host identifica la destinazione di rete alla quale viene indirizzata la richiesta.
Può essere rappresentato da un nome di dominio oppure da un indirizzo IP.

Esempio con un nome di dominio: https://example.com

Esempio con un indirizzo IPv4: http://192.0.2.10

Quando viene utilizzato un nome di dominio, il DNS permette al browser di ottenere l'indirizzo IP associato all'host specificato (Mockapetris, 1987).

**Porta**
La porta identifica il punto di accesso al servizio di rete con il quale il client deve comunicare. Gli schemi http e https hanno porte predefinite:

- Il protocollo associato HTTP utilizza normalmente la porta 80;
- Il protocollo associato HTTPS utilizza normalmente la porta 443 (IANA, n.d.-a).

Quando viene usata la porta predefinita, il numero può essere omesso.
I due URL seguenti fanno quindi riferimento alla stessa porta:

- https://example.com
- https://example.com:443

Una porta diversa deve invece essere indicata esplicitamente per esempio "https://example.com:8443"

**Percorso**
Il percorso, o path, identifica un obiettivo all'interno dello spazio di indirizzamento gestito dal server o dall'applicazione (Berners-Lee et al., 2005).

Per esempio: /catalogo/scarpe

Il percorso può assomigliare a una struttura di cartelle e sottocartelle, ma non corrisponde necessariamente a cartelle o file realmente presenti nel filesystem del server.

In un sito statico, un percorso può corrispondere direttamente a un file.
Per esempio: /immagini/logo.svg

In un'applicazione dinamica, invece, un percorso come "/catalogo/scarpe" può essere interpretato dal software lato server come un'istruzione per recuperare alcuni dati e generare la risposta.

Il percorso identifica quindi l'obiettivo della richiesta, ma il suo significato concreto viene stabilito dal server o dall'applicazione (Nottingham, 2020).
Negli URL con schema http e https, quando non viene scritto esplicitamente un percorso, viene utilizzato il percorso radice "/" dopo la dichiarazione dell'autorità.

In questo caso è il server che decide quale contenuto associare a questo percorso.
In un sito statico può essere configurato per restituire un documento come index.html, ma questa è una configurazione del server e non una regola universale degli URL.

***Esempio***

https://example.com/
In questo URL non viene specificato il percorso; nel server sarà quindi presente una regola che associa l'assenza del percorso a una particolare risorsa web.

***Esempio***
https://example.com/prodotti/scarpe
In questo URL:

- Nel file system del server potrebbe essere presente una cartella prodotti e, al suo interno, una cartella scarpe contenente un file index.html.
- Oppure potrebbe essere presente una regola che indica al sistema di cercare nel database tutti i prodotti appartenenti alla categoria "scarpe" e di costruire una pagina con i risultati.

**Query string (opzionale)**
La query, spesso chiamata query string, è la parte *dell'URL* introdotta dal simbolo "?". Permette di aggiungere informazioni o parametri alla richiesta (Berners-Lee et al., 2005).

Esempio: https://example.com/catalogo?colore=nero&taglia=42

Una convenzione molto comune è utilizzare coppie formate dal nome di un parametro e dal relativo valore:
?parametro1=valore1&parametro2=valore2

Il carattere = separa il nome dal valore, mentre & separa le diverse coppie.
Questa è una convenzione molto diffusa, ma il significato e la struttura della query vengono stabiliti dall'applicazione (Nottingham, 2020).

Nell'esempio precedente i parametri potrebbero indicare che il catalogo deve essere filtrato mostrando prodotti di colore nero e di taglia 42.

La query può essere utilizzata per:

- Filtrare un elenco.
- Eseguire una ricerca.
- Scegliere una lingua.
- Ordinare i risultati.
- Indicare una pagina.
- Conservare o condividere una parte dello stato dell'interfaccia.

È importante specificare che quando viene effettuata una richiesta HTTP o HTTPS al server, la query viene normalmente inclusa nella richiesta inviata (Berners-Lee et al., 2005; Fielding et al., 2022).
Tale richiesta può essere elaborata dal server oppure, in alcune applicazioni, da un programma JavaScript eseguito nel browser.

***Esempio***
Il seguente URL appartiene a un servizio di albo pretorio online:
https://web.spaggiari.eu/sdg/app/default/albo_pretorio.php?sede_codice=MIIT0027

La parte “sede_codice=MIIT0027” è la query string.
Contiene un parametro chiamato sede_codice, al quale è assegnato il valore MIIT0027.

L’applicazione riceve questo parametro e può utilizzarlo per determinare quale albo pretorio mostrare.
Per esempio, potrebbe usarlo per recuperare i dati associati alla sede indicata e generare la risposta da inviare al browser.

**Frammento**
Il frammento è la parte dell’URL introdotta dal carattere #:
Esempio:
https://example.com/manuale#installazione

Il frammento permette di identificare una parte o una vista specifica della risorsa indicata dal resto dell’URL (Berners-Lee et al., 2005).

A differenza della query, il frammento non viene inviato al server.
Viene separato dal resto dell’URL e interpretato dal browser o dall’applicazione lato client (Berners-Lee et al., 2005).
In un documento HTML, il frammento può corrispondere a un elemento dotato di uno specifico attributo id:

Facciamo finta che nel nostro documento HTML ci sia questo pezzo di codice: <h2 id="installazione">Installazione</h2>

Quando richiediamo al server questo URL: https://example.com/manuale#installazione

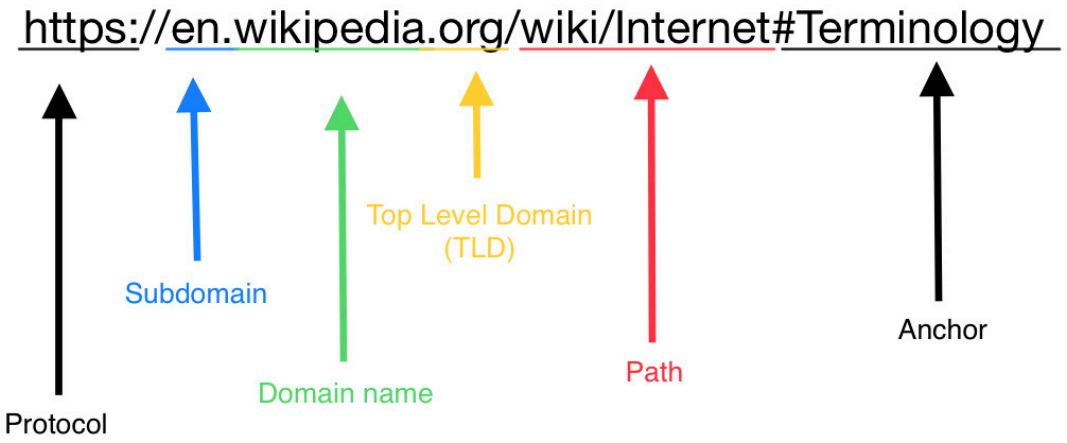
il browser scarica la risorsa /manuale, cerca l’elemento con id="installazione" e generalmente lo porta nell’area visibile (comportamento particolarmente utile nelle pagine molto lunghe, in cui l’utente non vuole scorrere fino alla sezione desiderata) (WHATWG, 2026).

Nel linguaggio comune, un collegamento contenente un frammento viene spesso chiamato **link di ancoraggio** (Berners-Lee et al., 2005).
Più precisamente, però:

- Il frammento è la componente dell’URL;
- L’ancora di destinazione è il punto o l’elemento individuato all’interno del documento.

**Esempio**
La pagina di Wikipedia dedicata a Internet è molto lunga ed è suddivisa in diverse sezioni. Il seguente URL contiene un frammento che permette al browser di portare direttamente alla sezione “Internet nei primi anni 2000”:
https://it.wikipedia.org/wiki/Internet#Internet_nei_primi_anni_2000

La parte “#Internet_nei_primi_anni_2000” è il frammento dell’URL.
In una pagina HTML, il browser cerca normalmente un elemento il cui attributo id corrisponde al testo che segue il simbolo #, e, se lo trova, porta quella parte del documento nell’area visibile.



# Materiale di supporto

## Esercizi

**Internet e Web sono la stessa cosa?**
Uno studente sostiene:
«Internet e World Wide Web sono due nomi diversi per indicare la stessa cosa.»
Un altro risponde:
«No, Internet è la rete, mentre il Web è uno dei sistemi che utilizza quella rete.»

Analizza le due affermazioni.

- Spiega quale delle due descrizioni è più corretta secondo la dispensa.
- Descrivi quale funzione svolge Internet.
- Descrivi quale funzione svolge invece il World Wide Web.
- Spiega perché il Web può essere considerato un sistema distribuito.
- Indica almeno tre tipi di risorsa che possono essere disponibili attraverso il Web.
- Immagina che Internet continui a funzionare ma tutti i web server diventino temporaneamente irraggiungibili. Spiega perché questo non significherebbe necessariamente che l'intera Internet abbia smesso di funzionare.
- Spiega infine perché il browser deve utilizzare Internet per raggiungere una risorsa web, ma Internet non coincide con il browser né con il Web.

**Costruire una navigazione ipertestuale**
Una scuola vuole pubblicare sul proprio sito una guida per i nuovi studenti.
La guida contiene:
Home → Presentazione della scuola → Indirizzi di studio → Laboratori → Regolamento → Contatti

La scuola vuole però evitare che gli studenti siano obbligati a leggere le pagine in questo ordine. Vuole inoltre permettere di:

- Passare direttamente dalla pagina di un indirizzo di studio ai relativi laboratori.
- Aprire un documento PDF contenente il regolamento.
- Passare dalla pagina dei contatti al sito dell'azienda dei trasporti.
- Saltare direttamente alla sezione "Orari" presente nella stessa pagina dei contatti.

Progetta concettualmente la struttura:

- Spiega perché la guida può essere organizzata come un ipertesto.
- Individua almeno quattro collegamenti ipertestuali che inseriresti.
- Per ciascuno, individua origine e destinazione.
- Distingui un collegamento verso un'altra pagina dello stesso sito, uno verso un sito esterno e uno verso una sezione della stessa pagina.
- Spiega perché l'origine di un link non deve necessariamente essere una parola.
- Se nella guida vengono collegati anche video, immagini e audio, spiega perché possiamo parlare di ipermedia.
- Spiega perché una struttura ipertestuale permette all'utente di costruire percorsi di navigazione differenti anziché seguire necessariamente una sequenza prestabilita.

**Che cosa succede quando apro un e-commerce?**
Uno utente apre un negozio online e visita la pagina:
https://shop.example.com/catalogo

La pagina mostra il logo dell'azienda, il foglio di stile del sito e un catalogo di prodotti che cambia in base alla disponibilità presente nel database.
Ricostruisci ciò che avviene.

- Spiega quale ruolo svolge il browser.
- Spiega quale ruolo svolge il web server.
- Indica quale protocollo viene utilizzato principalmente nello scambio di richieste e risposte tra browser e server.
- Spiega perché oggi viene normalmente utilizzato HTTPS.
- Classifica il logo, il file CSS e un eventuale file JavaScript già conservato sul server come risorse statiche o dinamiche.
- Spiega perché un file JavaScript può essere statico dal punto di vista del server ma produrre comunque comportamenti dinamici nel browser.
- Spiega perché la pagina del catalogo può invece essere considerata una risorsa dinamica.
- Ricostruisci il possibile processo: richiesta del browser → web server → applicazione lato server → database → generazione della risposta → browser.
- Spiega perché la pagina visualizzata dall'utente può essere costruita utilizzando contemporaneamente HTML, CSS, JavaScript, immagini e dati generati dinamicamente.

**Web server, server fisico o VPS?**
Una piccola azienda vuole pubblicare il proprio sito.
Acquista una VPS, installa Linux e successivamente installa Nginx. Sulla stessa VPS vengono inoltre eseguiti un'applicazione lato server e un database.
Un dipendente afferma:
«La VPS è il web server.»

Analizza l'affermazione.

- Spiega che cos'è una VPS.
- Spiega che cos'è invece un web server secondo il significato principale utilizzato nella dispensa.
- Individua quale ruolo svolge Nginx nell'esempio.
- Spiega perché VPS e web server non sono sinonimi.
- Spiega come più VPS possano essere ospitate dallo stesso computer fisico.
- Indica quali risorse possono essere assegnate separatamente a una VPS.
- Ricostruisci il percorso di una richiesta: utente → Internet → VPS → Nginx → eventuale applicazione → risposta.
- Spiega perché una stessa VPS può ospitare contemporaneamente web server, applicazione e database senza che questi termini indichino la stessa cosa.

**Dove deve essere eseguito il programma?**
Devi progettare tre applicazioni web.

- Applicazione A: un semplice editor di pixel art. Il disegno deve reagire immediatamente al movimento del mouse e può essere salvato sul dispositivo.
- Applicazione B: un servizio bancario. Deve controllare saldo, identità dell'utente, permessi e trasferimenti di denaro.
- Applicazione C: un editor collaborativo. Alcune operazioni vengono eseguite localmente, ma documenti e account devono essere sincronizzati tra più dispositivi.

Per ciascuna applicazione:
- Stabilisci se sarebbe più adatta un'architettura prevalentemente lato client, prevalentemente lato server oppure ibrida.
- Motiva la scelta.
- Indica quali operazioni faresti eseguire al browser.
- Indica quali operazioni affideresti invece al server.
- Spiega perché nell'applicazione A può essere vantaggioso eseguire molte elaborazioni direttamente sul dispositivo.
- Spiega perché nell'applicazione B il browser non dovrebbe poter decidere autonomamente se un trasferimento bancario è autorizzato.
- Spiega perché il codice JavaScript inviato al browser non dovrebbe contenere informazioni segrete.
- Spiega perché l'applicazione C rappresenta bene il concetto di applicazione ibrida.
- Concludi spiegando perché "lato client" e "lato server" descrivono soprattutto dove viene eseguita l'elaborazione, e non due categorie completamente separate di siti.

**Smontare un URL**
Considera questo URL:
https://shop.example.com:8443/catalogo/scarpe?colore=nero&taglia=42#modelli

Analizzalo completamente.
- Individua schema, host, porta, percorso, query e frammento.
- Spiega quale funzione svolge lo schema https.
- Spiega quale funzione svolge l'host shop.example.com.
- Descrivi cosa deve accadere se il browser conosce il nome dell'host ma non il relativo indirizzo IP.
- Spiega perché la porta 8443 deve essere scritta esplicitamente, mentre normalmente la porta 443 potrebbe essere omessa in un URL HTTPS.
- Spiega perché /catalogo/scarpe non deve necessariamente corrispondere a due cartelle realmente presenti sul disco del server.
- Spiega come un'applicazione dinamica potrebbe interpretare quel percorso.
- Interpreta i parametri colore=nero e taglia=42.
- Spiega quale differenza esiste tra il simbolo ? e il simbolo #.
- Indica quale parte tra query e frammento viene normalmente inviata al server.
- Spiega cosa potrebbe fare il browser con il frammento #modelli.

Immagina ora di cambiare solamente il frammento:
https://shop.example.com:8443/catalogo/scarpe?colore=nero&taglia=42#prezzi

Spiega perché il browser potrebbe non avere bisogno di richiedere al server una risorsa differente.

# Bibliografia


Berners-Lee, T. (1990). *Information management: A proposal*. CERN.

Berners-Lee, T., Fielding, R., & Masinter, L. (2005). *Uniform Resource Identifier (URI): Generic syntax* (RFC 3986). Internet Engineering Task Force.

CSS Working Group. (2026). *CSS snapshot 2026*. World Wide Web Consortium.

Ecma International. (2026). *ECMAScript 2026 language specification* (ECMA-262, 17th ed.).

Fielding, R., Nottingham, M., & Reschke, J. (Eds.). (2022). *HTTP semantics* (RFC 9110). Internet Engineering Task Force.

Internet Assigned Numbers Authority. (n.d.-a). *Service name and transport protocol port number registry*. Retrieved August 27, 2026.

Internet Assigned Numbers Authority. (n.d.-b). *Uniform Resource Identifier (URI) schemes*. Retrieved August 27, 2026.

Mockapetris, P. (1987). *Domain names—concepts and facilities* (RFC 1034). Internet Engineering Task Force.

Nottingham, M. (2020). *URI design and ownership* (RFC 8820). Internet Engineering Task Force.

OWASP Foundation. (2021). *A01:2021—Broken access control*.

Scarfone, K., Souppaya, M., & Hoffman, P. (2011). *Guide to security for full virtualization technologies* (NIST SP 800-125). National Institute of Standards and Technology.

WHATWG. (2026). *HTML Living Standard*.

W3C Web Applications Working Group. (2026). *Service workers*. World Wide Web Consortium.


# Approfondimenti

- James F. Kurose e Keith W. Ross, Reti di calcolatori e Internet. Un approccio top-down, 8ª ed., Pearson, 2022 — Approfondisce il funzionamento delle applicazioni Internet e del Web dal punto di vista delle reti, con particolare attenzione all'architettura client-server, HTTP, DNS, socket e comunicazione tra browser e server. È particolarmente utile per comprendere ciò che avviene "dietro" una richiesta web e collegare il Web ai livelli sottostanti della rete.
- Andrew S. Tanenbaum, Nick Feamster e David J. Wetherall, Reti di calcolatori, 6ª ed., Pearson, 2023 — Offre una trattazione più ampia dell'architettura di Internet e del livello applicativo, approfondendo World Wide Web, HTTP, DNS, client e server e il rapporto tra applicazioni distribuite e infrastruttura di rete.
- Jon Duckett, HTML & CSS. Progettare e costruire siti web, Apogeo, 2017 — Approfondisce in modo visuale HTML e CSS, spiegando come vengono strutturati i documenti web e come contenuti, immagini, collegamenti, tipografia e layout vengono trasformati nelle pagine interpretate dal browser. È particolarmente adatto per sviluppare la parte della dispensa dedicata alle tecnologie fondamentali del Web.
- Jon Duckett, JavaScript & jQuery. Sviluppare interfacce web interattive, Apogeo, 2017 — Approfondisce la componente dinamica eseguita nel browser, mostrando come JavaScript possa modificare le pagine, reagire alle azioni dell'utente,

comunicare con servizi esterni e costruire interfacce web interattive. È quindi utile per sviluppare il concetto di elaborazione lato client introdotto nella dispensa.

- Jon Duckett, PHP & MySQL. Sviluppare applicazioni web lato server, Apogeo, 2022 — Approfondisce invece il lato server delle applicazioni web, spiegando come generare pagine dinamiche, ricevere richieste, elaborare dati, interrogare database e produrre risposte personalizzate. È particolarmente pertinente alla distinzione della dispensa tra risorse statiche, risorse dinamiche e applicazioni prevalentemente lato server.

# Nota editoriale

Per la realizzazione di questi materiali ho utilizzato strumenti di intelligenza artificiale generativa come supporto alla scrittura, in particolare per migliorare la forma del testo, riorganizzare i contenuti, correggere la formulazione e velocizzare alcune attività redazionali.

Lavorando autonomamente alla produzione di questi materiali, cerco infatti di automatizzare tutte le attività che possono esserlo, così da poter dedicare più tempo alla ricerca, alla progettazione e allo sviluppo dei contenuti.

L'intelligenza artificiale non determina però i contenuti dell'opera: la scelta degli argomenti, la struttura, le idee, le interpretazioni, gli esempi e l'impostazione didattica sono elaborati da me. L'IA viene quindi utilizzata principalmente come strumento di supporto alla produzione e alla revisione formale, mentre la responsabilità autoriale e progettuale dei contenuti rimane mia.

Salvo diversa indicazione, questo materiale è distribuito con licenza Creative Commons Attribuzione – Non commerciale – Condividi allo stesso modo 4.0 Internazionale (CC BY-NC-SA 4.0).

È quindi possibile condividere, ridistribuire, modificare e creare opere derivate a partire dal materiale, a condizione di attribuirne correttamente la paternità, di non utilizzarlo per scopi commerciali e di distribuire eventuali versioni modificate o derivate con la stessa licenza.

