**HTTP - LAB**

**CGI (Common Gateway Interface)**

* **Interfata intre serverul web si programe externe**
* Clientul trimite o cerere
* Serverul web detecteaza ca cererea vizeaza un program .cgi
* Serverul ruleaza programul cgi, ii transmite datele necesare – STDIN
* Programul CGI proceseaza cerea
* CGI returneaza un raspuns pe STDOUT
* Serverul web trimite acel răspuns înapoi către client ca parte a răspunsului

**HTTP - niv. Aplicație**

- **stateless** - nu păstrează informații între cereri.

- fiecare cerere HTTP este independentă – serverul nu știe ce ai cerut anterior.

- soluții precum cookie-uri, session-uri, și token-uri sunt folosite pentru a menține „starea”

- **client – server** - clientul initializeaza cereri : serverul raspunde cu date cerute

- **HTTP request + HTTP response**

- o cerere HTTP contine**:** Metoda(GET, POST), URL, Headers, Body\*

- un raspuns HTTP contine: Cod de status (ex: 200 OK), Headers, Body

- **HTTPS** **:** - versiunea **securizata** a HTTP

- foloseste TLS/SSL (protocoale) pt a cripta comunicatia intre client si server

- asigura**:** confidentialitate, integritate, autentificare

- **Metodele**: - GET: obtine date

- POST: trimite date (ex: formularele)

- PUT: inlocuieste complet o resursa

- PATCH: modifica partial o resursa

- DELETE: sterge o resursa

- HEAD: ca GET, dar fara body (doar headers)

- OPTIONS: afla ce metode sunt suportate pe server

- **Status codes**:

**1xx**- Informații, ex: 100 Continue

**2xx** – Succes, ex:

* + 200 OK – succes normal
  + 201 Created – resursa creata
  + 204 No Content – succes, dar fara continut

**3xx** – Redirectionare, ex:

* + 301 Moved Permanently
  + 302 Found,
  + 304 Not Modified
  + clientul trebuie sa faca o alta cerere catre o alta adresa.

**4xx** - Erori client: cererea este incorecta

* + 400 Bad Request
  + 401 Unauthorized
  + 403 Forbidden
  + 404 Not found
  + 405 Method not allowed

**5xx** - Erori la server : serverul a esuat in procesarea cererii

* + 500 Internal Server Error
  + 502 Bad Gateway
  + 503 Service Unavailable

**Cache-ul** este o **memorie temporară** în care se stochează **datele** care au fost deja accesate sau procesate, pentru a le putea accesa mai rapid în viitor, fără a mai necesita o nouă solicitare către sursa originală (de exemplu, serverul web sau baza de date).

**De exemplu**, când accesezi o pagină de știri, browserul poate salva (în cache) imaginile, fișierele CSS și JavaScript asociate acelei pagini. Astfel, dacă revii pe pagină sau navighezi pe una similară, browserul va folosi datele din cache pentru a o încărca mai rapid, fără să le mai ceară de la server.

- **Browser cache + Proxy cache**

**- Browser cache**: păstrează local resurse (imagini, pagini) pentru a le încărca mai rapid data viitoare.

**-**  **Proxy cache**: cache intermediar între client și server (ex: servere proxy sau CDN-uri) care ajută la reducerea încărcării serverului.

- **Cacheable methods : - GET** (de obicei), **HEAD ( pot fi stocate in cache )**

**- POST** poate fi cache-uit în anumite condiții stricte, dar rareori este.

- **Safe methods:** Metode care nu ar trebui să modifice starea de pe server:

* **GET**
* **HEAD**
* **OPTIONS**

- Ele sunt considerate „sigure” deoarece doar citesc date.

- **Idempotent methods**: Metode care, indiferent de câte ori sunt apelate, **au același efect**:

- **GET**

- **PUT**

- **DELETE**

- **HEAD**

- **OPTIONS**

Ex: un DELETE pe aceeași resursă de mai multe ori o va șterge o singură dată.

- **HTTP headers**

Informații suplimentare transmise în cerere sau răspuns.

Exemple:

* **Content-Type**: tipul conținutului (ex: application/json)
* **Authorization**: pentru autentificare
* **Cache-Control**: controlul cache-ului
* **User-Agent**: cine face cererea (ex: browserul)

- **URL: Uniform Resource Locator:** Adresa completă a unei resurse de pe internet.

Domain.com/path?queryString=val1&queryStr=val2#fragment

- **scheme** – Protocolul folosit (ex: http, https, ftp).

- **domain** – Numele domeniului sau IP-ul serverului (ex: example.com).

- **port** (opțional) – Portul serverului (ex: :80 pentru HTTP, :443 pentru HTTPS).

- **path** – Calea resursei (ex: /products/123).

- **queryString** (opțional) – Parametrii de interogare, de obicei în format key=value(ex: ?id=5&name=example).

- **fragment** (opțional) – Partea din URL care face referire la o secțiune anume a paginii (ex: #section1).

- **Special characters in URL:** sunt folosite pentru a separa sau codifica părți ale unui URL:

* **?** : Separă **query string** de calea URL-ului.
* **&** : Separă **parametrii de interogare** (key-value).
* **=** : Folosit pentru a asocia parametri de interogare cu valori.
* **#** : Indică un **fragment** al paginii (pentru a naviga într-o secțiune specifică).
* Exemple de caractere care trebuie **encodate** (pentru a nu cauza conflicte):
  + **Space** → %20
  + **#** → %23
  + **&** → %26

URL-urile **corect encode** vor asigura că toate caracterele speciale sunt utilizate corespunzător.

**HTTP Cookies**

**Un cookie HTTP este o mică bucățică de date stocată pe dispozitivul utilizatorului de către un server web. Aceste date sunt transmise între client (browser) și server în fiecare cerere.**

**Utilizare:**

**Cookie-urile sunt utilizate pentru a reține informații despre utilizator între cereri HTTP, pentru că protocolul HTTP este stateless (nu "ține minte" nimic între cereri). Ele permit personalizarea experienței, gestionarea sesiunii și urmărirea activității utilizatorului. Mai jos sunt cele mai comune utilizări:**

**Exemplu:**

**Când te loghezi pe un site (de exemplu, Gmail sau Facebook), serverul generează un token de sesiune unic pentru tine și îl trimite înapoi către browser sub forma unui cookie. Acest cookie este apoi trimis automat de browser la fiecare solicitare către site, astfel încât serverul știe că ești deja autentificat și nu îți cere să te loghezi din nou la fiecare pagină.**

**Tipuri de Cookies:**

1. **Session Cookies:**
   * **Sunt stocate temporar și șterse atunci când închizi browserul.**
   * **Utilizate pentru a menține starea între paginile unui site (ex: autentificare).**
2. **Persistent Cookies:**
   * **Sunt stocate pentru o perioadă definită și nu sunt șterse la închiderea browserului.**
   * **Utilizate pentru a păstra preferințele utilizatorului între sesiuni (ex: limba preferată).**
3. **Third-Party Cookies:**
   * **Setate de site-uri terțe (ex: platforme de publicitate sau social media).**
   * **Pot fi folosite pentru urmărirea utilizatorilor între diferite site-uri.**
4. **Secure Cookies:**
   * **Sunt transmise doar printr-o conexiune securizată (HTTPS).**
5. **HttpOnly Cookies:**
   * **Nu pot fi accesate prin JavaScript, doar prin cereri HTTP. Utilizate pentru securitatea sesiunilor.**

**Setarea unui cookie:**

* **Serverul trimite un header Set-Cookie în răspunsul HTTP:**

**Set-Cookie: userID=12345; expires=Wed, 21 Oct 2025 07:28:00 GMT; path=/; Secure; HttpOnly**

**Structura unui Cookie:**

* **Un cookie tipic conține:**
  + **Nume (ex: userID)**
  + **Valoare (ex: 12345)**
  + **Data expirării (opțional)**
  + **Căile la care cookie-ul este valid (ex: /, path=/admin)**
  + **Domeniul la care este valabil (ex: .example.com)**
  + **Securitate (opțional) – Dacă este folosit doar prin HTTPS (Secure), accesibil doar prin HTTP (HttpOnly), etc.**

**Atacuri legate de Cookies:**

1. **Cross-Site Scripting (XSS):**
   * **Atacatorii pot injecta scripturi în paginile web pentru a fura cookies (de obicei, dacă HttpOnly nu este setat).**
2. **Cross-Site Request Forgery (CSRF):**
   * **Atacatorii pot forța utilizatorii să trimită cereri neautorizate la un site pe care sunt autentificați, folosind cookies.**
3. **Cookie Theft:**
   * **Atacatorii pot intercepta cookies sensibile prin atacuri de tip man-in-the-middle (fără HTTPS) sau prin vulnerabilități de securitate.**
4. **Cookie Poisoning:**
   * **Modificarea manuală a valorilor cookie-urilor pentru a manipula aplicația.**

**Alternative la Cookies:**

1. **LocalStorage/SessionStorage:**
   * **Stocarea datelor în browserul utilizatorului, similar cu cookies, dar fără a le trimite la server la fiecare cerere.**
   * **LocalStorage: persista pe termen lung.**
   * **SessionStorage: este șters când sesiunea de browser se închide.**
2. **Token-based Authentication:**
   * **Folosirea tokenurilor JWT (JSON Web Tokens) pentru autentificare, care nu necesită stocarea sesiunilor pe server.**
3. **Fingerprinting:**
   * **Tehnica de a identifica un utilizator pe baza caracteristicilor dispozitivului său (ex: rezoluția ecranului, pluginuri ale browserului).**

**Cache = chestii de „view” (stil, structură, răspunsuri frecvente).**

**Cookies = starea utilizatorului (autentificare, preferințe).**

**Diferente**

**Cache-ul**

Este folosit în principal pentru **optimizarea performanței** și stochează date temporare care:

* **Nu sunt sensibile** (în mod normal).
* Se pot refolosi **fără riscuri** de securitate.

**Exemple de date din cache:**

* Aspectul unei pagini: fișiere CSS, JavaScript, imagini.
* Răspunsuri la interogări frecvente (ex: lista celor mai populare produse).
* Răspunsuri API care nu se schimbă des.

⚠️ Cache-ul **nu ar trebui să stocheze parole sau date sensibile**, tocmai din motive de securitate.

**Cookies**

Sunt folosiți pentru a **păstra starea** utilizatorului (autentificare, preferințe, etc).

**Exemple de date din cookies:**

* Token de autentificare (ca să rămâi logat).
* Numele utilizatorului (doar pentru afișare, nu parola!).
* ID-ul coșului de cumpărături.

⚠️ Parolele **nu se salvează nici în cookies**, ci doar identificatori sau token-uri (care sunt mai sigure și pot expira).