# Mini WEB Server



Indrumator:

Slt. Adina VAMAN

Realizat de:

MITRACHE Andreea

ROSCA George

Grupa C 113 C

Cuprins

[Mini WEB Server 1](#_Toc181173302)

[Introducere 3](#_Toc181173303)

[Scopul proiectului 3](#_Toc181173304)

[Lista definitiilor 3](#_Toc181173305)

[Structura documentului 4](#_Toc181173306)

[Arhitectura si componente SW 4](#_Toc181173307)

[Descrierea produsului software 4](#_Toc181173308)

[Detalierea platformei SW/HW 4](#_Toc181173309)

[Actori 4](#_Toc181173310)

[Arhitectura interna 5](#_Toc181173311)

[Descriere functionalitati 5](#_Toc181173312)

[Exemple de testare 6](#_Toc181173313)

[Snippet-uri 6](#_Toc181173314)

# Introducere

## Scopul proiectului

Proiectul urmărește dezvoltarea unui software capabil să funcționeze ca un server web simplu, responsabil de preluarea, interpretarea și gestionarea cererilor HTTP trimise de clienți, precum browserele web. Acest server va avea capacitatea de a primi cereri din partea clienților, cum ar fi solicitările de acces la o pagină web sau la o resursă specifică, și de a identifica tipul de conținut solicitat în fiecare cerere.

După recepționarea cererilor, serverul va analiza informațiile incluse în acestea, cum ar fi tipul resursei cerute și parametrii din antetul cererii, și va răspunde corespunzător, livrând conținutul necesar. Acest conținut poate consta în fișiere HTML pentru structura paginii web, fișiere CSS pentru stilizare, fișiere JavaScript pentru funcționalitate dinamică, imagini și alte tipuri de fișiere media sau fișiere de date (JSON, XML etc.), în funcție de solicitările clientului.

Proiectul își propune să creeze un server care să respecte standardele de comunicare prin HTTP, inclusiv metode de cerere cum ar fi GET, POST, și altele, pentru a asigura o experiență de utilizare conformă cu cerințele moderne. De asemenea, serverul va fi capabil să gestioneze erori comune și să ofere răspunsuri de eroare, cum ar fi pagini de eroare 404 pentru resurse inexistente, asigurând astfel o comunicare fiabilă și precisă între client și server.

## Lista definitiilor

**Porturi de rețea:** o valoare numerică asociată fiecărui protocol, utilizată pentru a facilita comunicarea necesară unui anumit serviciu sau funcție de rețea.

**IP (Internet Protocol):** protocol de comunicație care furnizează identificare unică și adresare pentru dispozitivele conectate la internet sau la o altă rețea de comunicații, asigurând astfel transmiterea corectă a datelor între dispozitive.

**Socket:** o metodă prin care se stabilesc și se gestionează legături între două puncte terminale (cum ar fi două dispozitive sau procesoare), facilitând schimbul de date. Un socket este format dintr-o pereche compusă din adresă IP și port.

**Thread (fir de execuție):** o unitate de execuție dintr-un proces; este responsabilă de rularea independentă a codului în cadrul procesului. Un proces reprezintă un program aflat în execuție într-un sistem de operare, iar un thread este o subunitate a acestui proces, care rulează simultan cu alte thread-uri în cadrul aceluiași program.

## Structura documentului

Documentul este împărțit în cinci capitole. Capitolul 1 reprezintă introducerea. Capitolul 2 prezintă actorii (tipurile de utilizatori ai aplicației) și arborele de fișiere al aplicației. Capitolul 3 prezintă funcționalitățile pe care aceasta le-o oferă utilizatorului. Capitolul 4 cuprinde exemple de testare a funcționalităților aplicației. Capitolul 5 prezinta snippet – uri.

# Arhitectura si componente SW

## Descrierea produsului software

Proiectul va fi realizat în limbajul de programare C și va transforma un dispozitiv într-un server web simplu, ușor de configurat și utilizat. Serverul va fi conceput pentru a oferi un mediu de dezvoltare și testare pentru aplicațiile web, permițând accesarea resurselor locale prin intermediul unei interfețe web accesibile din browser.

Aplicația va gestiona cererile HTTP de la clienți și va facilita accesul la fișiere locale, cum ar fi pagini HTML, stiluri CSS, scripturi JavaScript și alte resurse media. Acest server web simplificat permite dezvoltatorilor să testeze și să vizualizeze funcționalitățile aplicațiilor lor web într-un mediu local, eliminând necesitatea unor configurări complexe de rețea sau servere externe.

Astfel, aplicația va oferi un instrument eficient pentru accesarea resurselor locale și va facilita verificarea rapidă a funcționalităților și interacțiunilor într-un browser, oferind o experiență de testare rapidă și flexibilă pentru dezvoltatori.

## Detalierea platformei SW/HW

Acest software este destinat dispozitivelor ce rulează pe sistemul de operare Linux și include un server HTTP personalizat, scris în C, capabil să gestioneze și să răspundă eficient cererilor HTTP venite de la clienți. Serverul procesează fiecare cerere primită și trimite răspunsuri precise, conform protocolului HTTP.

Produsul oferă un sistem de configurare flexibil, prin care utilizatorii pot seta diferiți parametri ai serverului, cum ar fi portul de ascultare, directorul principal pentru resurse (rădăcina site-ului) și locația fișierelor de configurare. În plus, software-ul include mecanisme robuste de gestionare a erorilor, generând automat coduri de stare HTTP și furnizând răspunsuri adecvate în cazuri de eroare, pentru a asigura o experiență de utilizare stabilă și clară.

## Actori

**Utilizatorii finali:**

Utilizatorii finali sunt entitățile (persoane sau organizații) care accesează și utilizează conținutul furnizat de serverul web. Aceștia interacționează cu interfața web prin intermediul unui browser sau al unei aplicații dedicate, având acces la diverse resurse disponibile pe server, cum ar fi pagini HTML, fișiere multimedia, sau date procesate în timp real.

**Dezvoltatorii:**

Dezvoltatorii sunt persoanele responsabile pentru crearea, dezvoltarea, testarea și întreținerea software-ului serverului web. Rolul lor include programarea și optimizarea serverului pentru a asigura performanță și stabilitate, gestionarea actualizărilor și a noilor funcționalități, precum și monitorizarea securității. Aceștia asigură și mențin compatibilitatea serverului cu cerințele protocoalelor și cu diversele tipuri de conținut și dispozitive care accesează platforma.

**Clienții:**

Clienții reprezintă dispozitivele sau aplicațiile software care inițiază cereri HTTP către serverul web pentru a accesa resursele sau pentru a comunica cu serviciile web oferite. Aceștia pot fi reprezentanți de browsere, aplicații mobile sau alte programe client ce solicită diverse resurse (cum ar fi fișiere, date JSON sau API-uri) pentru a procesa informațiile pe care utilizatorii le vizualizează sau le utilizează direct.

## Arhitectura interna

1. Componenta de gestionare a cererilor HTTP:

* **Analiza cererilor:** Serverul web primește cereri HTTP de la clienți și procesează fiecare cerere pentru a extrage informațiile necesare, precum URL-ul solicitat, metoda HTTP utilizată (GET, POST, PUT etc.), anteturile cererii și orice date incluse în corpul acesteia. Această analiză preliminară este esențială pentru a înțelege și a interpreta corect ce resurse sunt solicitate și ce răspuns va oferi serverul.
* **Tratarea cererilor:** Pe baza informațiilor extrase, serverul decide cum să gestioneze fiecare cerere. În funcție de tipul cererii, serverul poate returna un fișier static (cum ar fi o pagină HTML sau o imagine), poate executa un script pentru a genera conținut dinamic sau poate trimite cererea către o aplicație web sau un serviciu specific. Acest proces de tratament asigură că fiecare cerere este direcționată către resursa sau serviciul corespunzător.
* **Răspunsul la cereri:** După ce cererea este procesată, serverul generează răspunsul HTTP corespunzător. Răspunsul include codul de stare (de exemplu, 200 pentru succes, 404 pentru resursă negăsită), anteturile de răspuns (ce pot conține informații precum tipul de conținut și data generării) și corpul răspunsului, care poate fi fie conținutul solicitat (un fișier HTML, de exemplu), fie un mesaj de eroare. Această componentă asigură o comunicare coerentă și standardizată între server și client.

1. Componenta de gestionare a conținutului:

* **Gestionarea fișierelor:** Serverul trebuie să poată accesa sistemul de fișiere al gazdei pentru a localiza și furniza fișierele solicitate, fie că sunt resurse statice (HTML, CSS, JavaScript) sau dinamice (scripturi PHP sau alte resurse server-side). Serverul trebuie să gestioneze eficient accesul la aceste fișiere pentru a evita conflicte și pentru a menține performanța.
* **Procesarea scripturilor:** Pentru a genera conținut dinamic, serverul poate suporta limbaje de programare server-side (precum PHP, JavaScript server-side, etc.) printr-o componentă de interpretare și execuție a scripturilor. Această componentă permite serverului să ruleze scripturi și să genereze conținut dinamic în funcție de cerințele specifice ale cererii, cum ar fi date personalizate sau informații generate pe baza parametrilor primiți de la client.

1. Componenta de configurare:

* **Citirea și interpretarea fișierelor de configurare:** Serverul poate citi fișiere de configurare care stabilesc parametri esențiali de funcționare, cum ar fi portul de ascultare, numărul de thread-uri, directorul rădăcină pentru resurse și alte setări personalizate. Acest proces de configurare permite utilizatorilor să personalizeze funcționarea serverului și să ajusteze resursele în funcție de cerințele aplicației. Fișierele de configurare sunt interpretate la inițializarea serverului și pot fi actualizate pentru a adapta serverul la noi cerințe sau scenarii de utilizare*.*

## Descriere functionalitati

1. *Pornire și Oprire a Serverului:*

Permite utilizatorilor să pornească și să oprească serverul web.

1. *Configurare a Serverului:*

Permite utilizatorilor să configureze parametrii serverului web, cum ar fi portul de ascultare, autentificare, numar clienti, și altele.

1. *Autentificare:*

Permite conectarea de catre utilizator la server.

1. *Interpretare GET text/html sau text/plain:*
2. *Interpretare GET image/jpeg sau image/png:*
3. *Interpretare GET php:*
4. *Interpretare GET JAVAScript:*
5. *Interpretare GET fisier neinterpretabil:*
6. *Interpretare GET fisier inexistent:*

# Exemple de testare

# Snippet-uri