**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare Informatică și Microelectronică**

**Departamentul** **Ingineria Software şi Automatică**

**Raport**

**Lucrare de laborator nr.1**

Disciplina: Programarea în rețea

Tema: Socket | HTTP | Multithreading

A îndeplinit: st.gr.TI-172 Serețan Evelina

A controlat: Buldumac Oleg

Chișinău 2020

Scopul lucrării:

1. Să se creeze o cerere(request) GET HTTP către web-serverul website-ului unite.md utilizînd socket.

2. Cu ajutorul expresiilor regulate să se extragă toate imaginile din raspunsul primit din punctul 1.

3. Utilizînd Socket, HTTP și firele de execuții să se descarce toate imaginile găsite din puncul 2 într-un folder

Intrebări de control:

1. Ce este un socket?
2. Un socket este un punct intern pentru trimiterea sau primirea datelor într-un nod dintr-o rețea de calculatoare. Concret, este o reprezentare a acestui punct final în software-ul de rețea, cum ar fi o intrare într-un tabel și este o formă de resursă de sistem.
3. Scopul protocolului HTTP?

Scopul protocolului HTTP este de transmiterea și obtinerea paginii web de la distanță.

1. Ce este un fir de execuție?

Un fir de execuție este cea mai mică unitate de procesare ce poate fi programată spre execuție de către sistemul de operare.

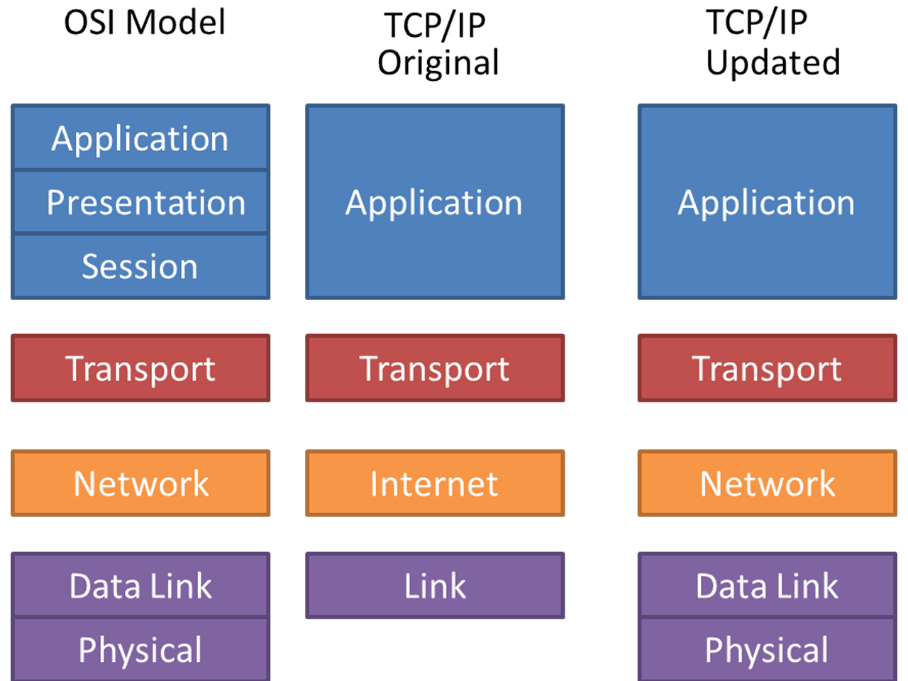
1. Diferența dintre TCP și UDP?

TCP așteaptă confirmare ca pachetele au fost primite, UDP nu. Respectiv, TCP urmarește pachetele, așa ca datele nu riscă de a fi pierdute, UDP nu le urmărește. UDP ne oferă însa rapiditate. De regula este folosit în chat-uri, video real time, ș.a.

1. Ordinea în care octeții sunt secvențializați în rețea?

Cea mai frecvent utilizată ordine este big-endian care plasează octeții de la cel mai semnificativ la mai puțin semnificativ.

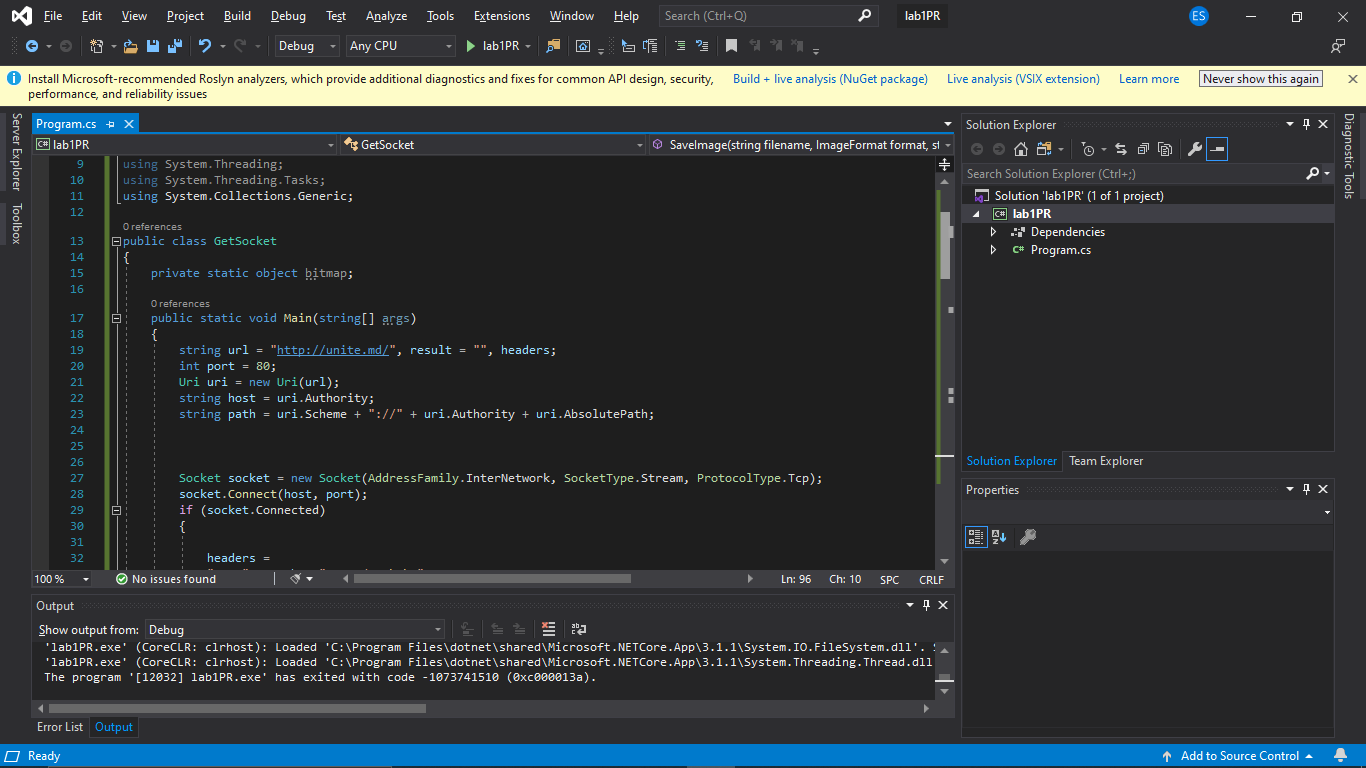
1. Cele patru nivele ale stivei TCP/IP și șapte nivele ale modelului OSI?



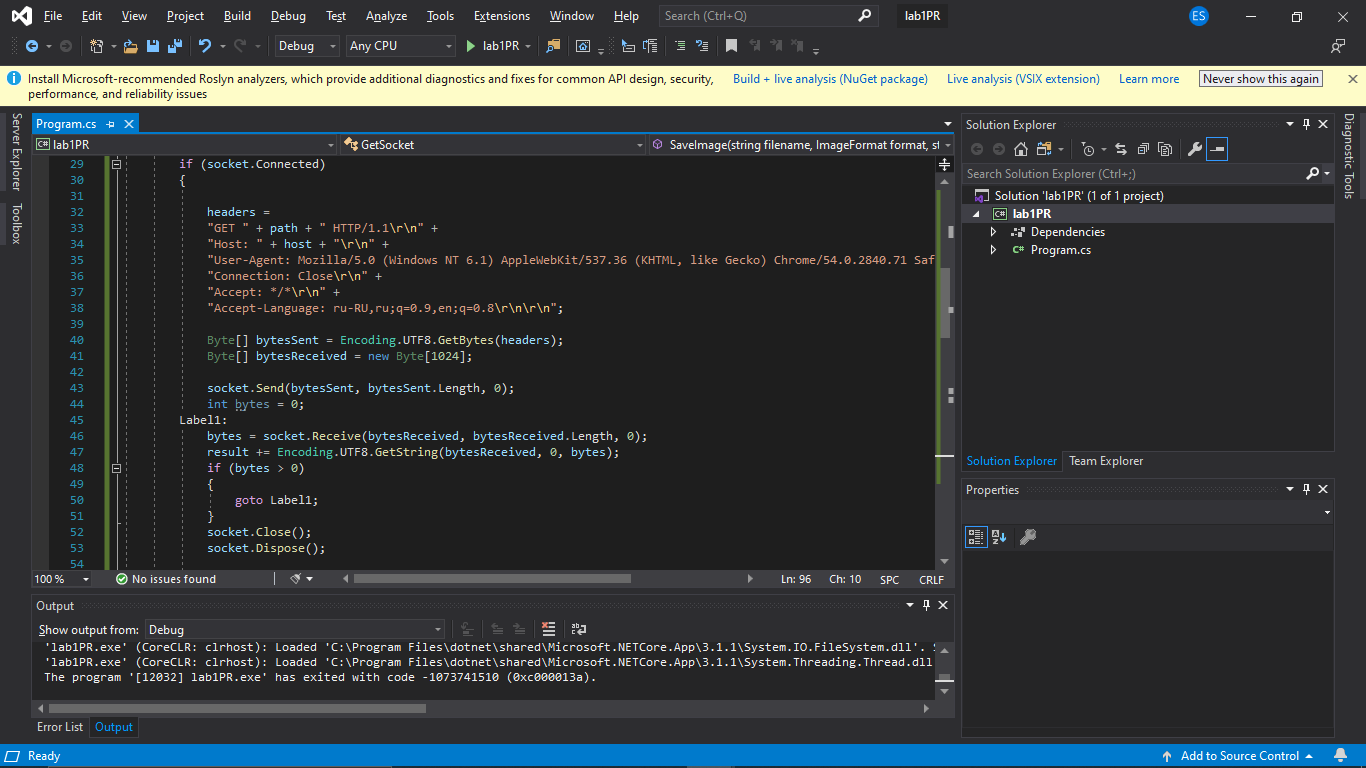
1. Din ce cauză nu puteți transmite cereri HTTP website-ului utm.md utilizînd portul 80 ?

Deoarece utm.md folosește protocolul HTTPS care se alfă pe alt port-443.

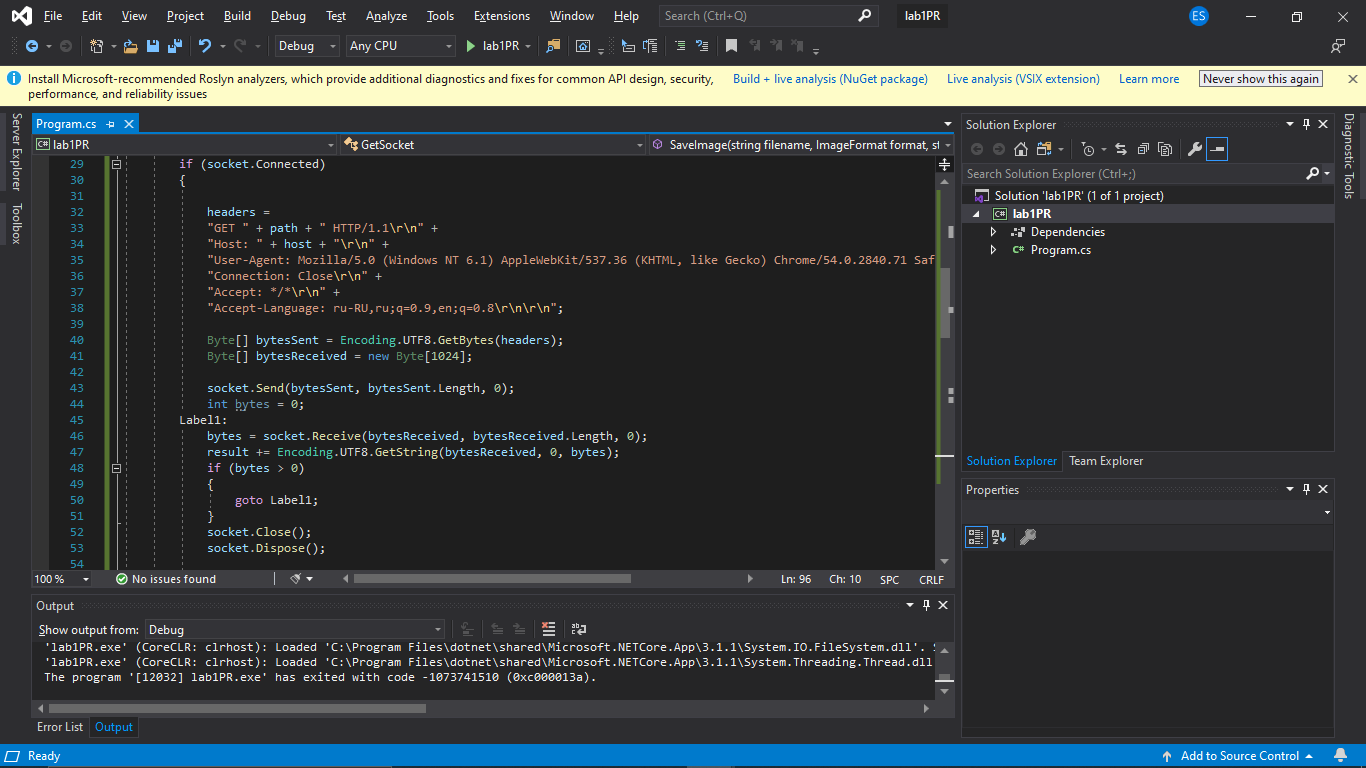
1. Cum se creează o conexiune Socket ?



1. Cum se transmit date intr-un Socket ?



1. Cum se citesc datele dintr-un Socket ?



12.Care sunt metodele HTTP?

Metodele disponibile sunt :

1. **GET**: este cea mai folosită metodă, fiind utilizată atunci când serverului i se cere o resursă.
2. **HEAD**: se comportă exact ca metoda GET, dar serverul returnează doar antetul resursei, ceea ce permite clientului să inspecteze antetul resursei, fără a fi nevoit să obțină și corpul resursei.
3. **PUT**: metoda este folosită pentru a depune documente pe [server](https://ro.wikipedia.org/wiki/Server), fiind inversul metodei GET.
4. **POST**: a fost proiectată pentru a trimite date de intrare către server.
5. **DELETE**: este opusul metodei PUT.
6. **TRACE**: este o metodă folosită de obicei pentru diagnosticare, putând da mai multe informații despre traseul urmat de legătura HTTP, fiecare server proxy adăugându-și semnătura în antetul Via.
7. **OPTIONS**: este folosită pentru identificarea capacităților serverului Web, înainte de a face o cerere.
8. **CONNECT**: este o metodă folosită în general de serverele intermediare.

13.Codurile de stare HTTP (200, 301, 302, 401, 404, 405, 500)

**200 - ok:**  
Această cerere a fost executată cu succes. Informația a revenit cu un răspuns pozitiv, indiferent de modul în care s-a făcut cererea

**301 - modificat/mutat permanent:**  
Cererii i-a fost atribuite o sursă nouă și permanentă URI iar cererile următoare ar trebui să folosească una din sursele returnate URI. Dacă acest mesaj/cod este primit ca răspuns al unei cereri tip „Primit/recepționat” sau „Direcționat/condus”, browser-ul nu trebuie să redirecționeze automat cererea, doar dacă nu poate fi confirmată de către utilizator.

**302 - găsit:**  
Sursa cererii este temporar pe un alt URI. În cazul în care redirectarea ar putea fi schimbată ocazional, utilizatorul ar trebui să folosească în continuare cererea URI (Request-URI) în cazul unor cereri ulterioare.

Dacă mesajul/statusul 302 este recepționat ca răspuns al unei cereri alta decât „Primit/recepționat” sau „Direcționat/condus”, browser-ul nu trebuie să redirecteze automat cererea dacă aceasta nu poate fi confirmată de cărte utilizator.

**401 - neautorizat:**  
Cererea necesită autentificarea/înregistrarea utilizatorului. Răspunsul trebuie să includă WWW - câmp de autentificare conținând o somație aplicabilă utilizatorului. Utilizatorul poate repeta cererea. Dacă cererea deja includea câmpul de autorizare, atunci raspunsul 401 indică faptul că autorizarea a fost refuzată. Dacă răspunsul 401 conține aceeași somație ca răspuns principal iar browser-ul a executat autentificarea cel puțin o dată, atunci utilizatorul ar trebui să i se prezinte intrarea dată în răspuns, din moment ce aceasta include informații relevante.

**404 - negăsit:**  
Serverul nu a găsit nimic care să corespundă cererii URI. Nu se dau indicații despre condiția temporară sau permanentă.

**405 - metodă nepermisă:**  
Metoda specificată în linia de cerere (Request-Line) nu este permisă de către sursa identificată după URI-ul cerut.

**500 - eroare internă de server:**  
Server-ul a întâmpinat o condiție neașteptată și o previne spre a putea îndeplini cererea.

14.Ce este HTTP securizat?

Secure Hyper Text Transfer Protocol reprezintă protocolul HTTP încapsulat într-un flux SSL/TLS care criptează datele transmise de la un browser web la un server web, cu scopul de a se oferi o identificare criptată și sigură la server.

15.Diferența dintre HTTP și HTTPS.

Site-urile care utilizează HTTPS sunt criptate. Asta inseamna ca toate informațiile private, precum nume utilizator, parole, adrese de email, adresa, informatiile despre cardurile dvs sunt pastrate in siguranta. La accesarea unui site care foloseste HTTPS, singurii care au acces la informatiile transmise sunt clientul si serverul,iar HTTP este protocolul care permite comunicarea cu site-urile dorite; este un protocol de tip text, fiind protocolul implicit al WWW (World Wide Web).

16.Diferența dintre Socket și WebSocket.

WebSockets rulează de obicei din browserele care se conectează la Application Server printr-un protocol similar cu HTTP care trece TCP / IP. Deci sunt pentru aplicații web care necesită o conexiune permanentă la serverul său. Pe de altă parte, socket-urile sunt mai puternice și generice. Se execută peste TCP / IP, dar nu sunt restricționate la browsere sau protocolul HTTP. Ele ar putea fi folosite pentru a implementa orice fel de comunicare.

17.Diferența dintre un proces și un fir de execuție.

[Procesul și firul sunt două tehnici folosite de programatori pentru a controla procesorul și executarea instrucțiunilor pe un computer într-o manieră eficientă. Un proces poate conține mai multe fire. Firele furnizează o modalitate eficientă de a partaja memoria, deși operează mai multe execuții decât procesele. Prin urmare, firele sunt o alternativă la mai multe procese. Odată cu tendința crescândă spre procesoarele cu mai multe nuclee, firele vor deveni cel mai important instrument din lumea programatorilor.](https://ro.betweenmates.com/difference-between-process-and-thread-9317#menu-5)

18.Ce este un Deadlock și cînd el apare?

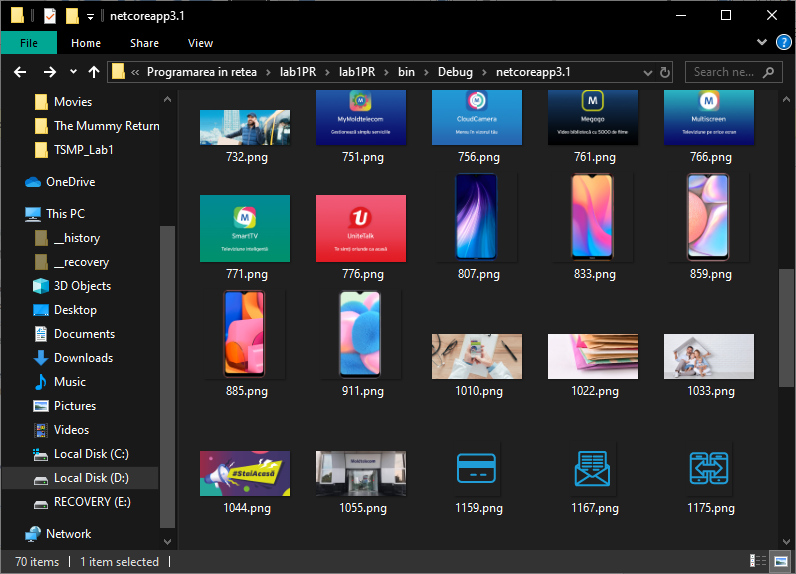
Deadlock este o situație în care un set de procese sunt blocate, deoarece fiecare proces deține o resursă și așteaptă o altă resursă dobândită de un alt proces. ... Situație similară apare în sistemele de operare când există două sau mai multe procese care dețin unele resurse și așteaptă resursele deținute de alte persoane.

19.Pentru ce este nevoie de metoda join() în programarea multithreading?

join (): va pune firul curent în așteptare până când firul pe care este numit este mort. Dacă firul este întrerupt, atunci va arunca InterruptException.

Dacă există mai multe fire care numesc metodele Join (), ceea ce înseamnă că supraîncărcarea pe unire permite programatorului să specifice o perioadă de așteptare. Cu toate acestea, la fel ca în cazul somnului, Join este dependentă de sistemul de operare pentru cronometrare, deci nu ar trebui să presupună că alăturarea va aștepta exact atâta timp cât se specifică.

Rezultate obținute:



Concluzie:

In urma efectuării acestei lucrări de laborator am avut ocazia să înțelegem cum se lucrează cu socket-urile. De asemenea au fost căpătate cunoștințe în ceea ce privește protocoalele HTTP, HTTPS, metodele acestor protocoale, metode de transmitere a pachetelor și altele.

În lucrarea dată s-a realizat conectarea la site-ul “unite.md” prin intermediul socket-ului, după care din codul HTML au fost extrase toate denumirile de imagini(\*.jpg;\*.png;\*.gif) intr-un array de string-uri prin intermediul expresiilor regulate, iar in final, toate aceste imagini au fost descărcate pe ”D:\Univer\Programarea in retea\lab1PR\lab1PR\bin\Debug”

În final, am analizat rezultatele și formulat concluzii.