

**UNIVERSITATEA DIN BUCUREȘTI**  
**FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ**

**Specializarea: Informatică**

# **LUCRARE DE LICENȚĂ**

Coordonator științific  
**Prof. Dr. Liviu DINU**

Absolvent  
**Teodor Mircea POPESCU**

**București**  
**2017**



**UNIVERSITATEA DIN BUCUREȘTI**  
**FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ**

**Specializarea: Informatică**

**Metode de Automatizare  
Creativă a Rutinelor  
Ordinare și Utilizarea lor  
(M.A.C.R.O.U.L.)**

Coordonator științific  
**Prof. Dr. Liviu DINU**

Absolvent  
**Mircea Teodor POPESCU**

**București**  
**2017**



# Cuprins

<b>INTRODUCERE.....</b>	<b>1</b>
<b>CAPITOLUL 1. DESCRIEREA GENERALĂ A PLATFORMEI.....</b>	<b>2</b>
<b>CAPITOLUL 2. APLICAȚIA "MACROUL" .....</b>	<b>3</b>
2.1. POVESTEA .....	3
2.2. ACTIVATORI .....	4
2.2.1 Comandă vocală: .....	4
2.2.2 Apăsarea unei combinații de taste: .....	4
2.2.3 Activarea macroului la un moment de timp ales: .....	4
2.2.4 Introducerea unui card RFID: .....	5
2.2.5 Activarea de pe o pagină web: .....	5
2.3. MODUL DE OPERARE .....	5
2.3.1. "Adaugare/Editare macro" .....	6
2.3.2. Editorul de macrouri .....	7
2.3.3. C (Click) .....	8
2.3.4. M (Move / Mută) .....	10
2.3.5. S (Sleep / Așteaptă) .....	11
2.3.6. K (Keyboard / Taste) .....	11
2.3.7. E (Execută) .....	13
2.3.8. W (Website) .....	14
2.3.9. @ (Email) .....	15
2.3.10. P (Power) .....	15
2.3.11. H (SSH) .....	16
2.3.12. X (Închide) .....	17
2.3.13. R (Resize / Redimensionare) .....	18
2.3.14. A (Așteaptă) .....	19
2.3.15. Q (Macrou / Sablon) .....	21
2.3.16. "Macrourele mele" (Activatori) .....	22
2.3.17. "Cauta macrou" .....	24
2.3.18. "Calculatoare Grupuri si Meniuri" .....	25
2.3.19. "Sabloane" (Macrouri cu variabile) .....	27
2.3.20. "Setari si informatii" .....	31
<b>CAPITOLUL 3. WEBSITE .....</b>	<b>33</b>
<b>CAPITOLUL 4. EXEMPLE .....</b>	<b>37</b>
<b>CAPITOLUL 5. PROVOCĂRI.....</b>	<b>41</b>
<b>CAPITOLUL 6. DIRECȚII DE ÎMBUNĂTĂȚIRE .....</b>	<b>43</b>

<b>CAPITOLUL 7. CONCLUZII.....</b>	<b>44</b>
<b>CAPITOLUL 8. LINK-URI UTILE.....</b>	<b>45</b>

# Introducere

Această lucrare de Licență a pornit de la un gând simplu pe care l-am avut cu mult timp în urmă, în școala generală când, copil fiind, am descoperit un joc video. În primele luni m-am lăsat purtat de joc într-o aventură nouă pentru mine dar, cu timpul, farmecul a început să se piardă și aventurile s-au schimbat în numeroase sarcini repetitive. Nu erau sarcini grele, dar simțeam că pentru a progresa trebuie să petrec prea mult timp în joc. Atunci a fost pentru prima dată când mi-am dorit să îi pot spune calculatorului ce să facă. Asta m-a inspirat să învăț să programez și pasiunea pentru programare se regăsește astăzi în alte forme, în alte locuri, dar în aceeași măsură și cel mai important, cu același ideal - eficiența.

Pe parcursul ultimilor ani am creat diverse funcții în diverse limbaje de programare, funcții simple pentru sarcini cotidiene - click-ul, deschisul ferestrelor, trimisul de mail-uri, închiderea calculatorului după un anumit timp, dar mai mereu trebuia să le refac pentru că se pierdeau printre alte proiecte. Îmi rămânea însă mereu întipărită în minte ideea că le-am făcut și știam exact cum să le refac, dar tot îmi lua timp. Când am ajuns la locul de muncă, m-am confruntat cu aceeași problemă - multe sarcini simple și repetitive. Astfel, a fost natural să mă gândesc să automatizez și aceste lucruri și mai mult, în anul III am automatizat împreună cu un coleg rezolvarea subiectelor de examen la Dezvoltarea Aplicațiilor Web. Aveam voie cu orice materiale vroiam, condiția era doar să nu comunicăm în timpul examenului, ceea ce nu am făcut. Astfel cu ajutorul programului menționat doar scriam câteva rânduri luate din cerințe, apăsam un buton și Website-ul era creat. Reușita acestui program mi-a reînnoit convingerea că până și în locuri în care nu ne așteptăm este nevoie de automatizări.

Pus în fața alegerii unei licențe care să mă reprezinte și care să fie utilă nu doar mie ci și altora (celor care poate sunt mai puțin înclinați către programare) am decis să fac o platformă minimalistă, fără dependențe greoaie, a cărei interfață să nu încerce să fie nici prea generică dar nici prea accesibilă, ci o combinație cât mai bună între cele două pentru a da un optim de expresivitate utilizatorului considerat non-programator. Mi-am propus să elimin nevoia acelor mici rutine pe care le tot scriam și pe care le tot vedeam scrise și rescrise pe stackoverflow, să găsesc acel nucleu care să facă posibilă rezolvarea rutinelor și de la care ele pot fi extinse sau compuse spre a fi un fel de LEGO al programatorului, o trusă de unelte care pot fi folosite pentru rezolvarea parțială sau chiar completă a multor probleme cotidiene.

Astfel, lucrarea va insista nu atât pe procedeul tehnic (din spate), ci pe cel non-tehnic (din față) și va încerca să susțină utilitatea aplicației și să ofere diferite cazuri de utilizare cât mai accesibile, care să îi pună în evidență valoarea. Îmi doresc ca la final, cititorul să își dorească să utilizeze această aplicație și să aibă în minte un scenariu pe care eu încă nu l-am prevăzut și care ar putea să îi fie util. Este totodată speranța mea că acesta va contribui la dezvoltarea unui sistem public de rutine, dacă se consideră că ele pot fi de folos și altora în căutarea unei astfel de rezolvări și că în acest fel activitatea umană va deveni mai eficientă.

# Capitolul 1. Descrierea generală a platformei

Platforma descrisă în această lucrare își propune să fie accesibilă și expresivă și să permită cât mai multor oameni să își rezolve problemele. Aplicația "Macroul" are o sintaxă extrem de simplă, este gratuită, este open-source și extensibilă de către terțe persoane, atât direct prin modificarea codului sursă, cât și indirect prin publicarea unor executabile utile care sunt accesibile "Macroului", prin comanda "Execută".

Platforma este una distribuită în sensul că fiecare entitate își poate de fini rețeaua proprie dacă are acces la un calculator care poate să fie folosit drept server (poate rula MySQL, PHP și este vizibilă de către calculatoarele care doresc să fie clienți). Astfel, datele rămân la nivelul acestei rețele și în acest caz securitatea poate fi asigurată local, iar ea se poate extinde atunci când serverul ajunge vizibil din alte rețele. Aceste condiții nu vizează în niciun sens clienții ce pot vedea serverul, nefiind deci necesar port-forwarding pentru aceștia. Tot ce trebuie să facă un utilizator este să deschidă aplicația (care nu trebuie nici măcar instalată) și poate beneficia de toate funcționalitățile acesteia.

Un alt sens în care platforma este distribuită este că permite orchestrarea unor acțiuni complexe pe mai multe mașini concomitent, cum ar fi instalarea în paralel a unui software la un moment bine determinat sau se pot da în prealabil date fiecărui calculator din rețea și acestea pot desfășura aceleași acțiuni, inclusiv trimiterea rezultatelor prelucrării spre a fi centralizate. Astfel, dacă prelucrarea poate fi făcută cu ajutorul Macroului, el poate fi folosit și pentru paralelizarea prelucrării.

Tot pe server sunt stocate macrourele utilizatorilor, deci domeniul de vizibilitate al celor făcute publice este exact domeniul de vizibilitate al serverului, deci se creează un sistem propriu, independent și dacă este vorba de unul global utilizatorii pot contribui la dezvoltarea platformei sau pot alege ca acțiunile lor să rămână cu totul private.

Arhitectura aplicației este concepută pe trei niveluri:

- Baza de date MySQL care stochează datele și informațiile de stare;
- Serverul PHP care face accesibilă respectiva informație atât clienților desktop cât și celor mobili;
- Clienții Desktop (sau Mobili) dintre care primii au capacitatea de a crea și utiliza macroure iar ceilalți au capacitatea de a apela macroure create în prealabil de diverși clienți la care li s-a dat acces sau pot accesa direct acele desktop-uri pentru o gestiune personalizată.

Aceasta transformă "Macroul" într-o unealtă puternică de acces la distanță, complet gratuită și a cărei viteză este limitată doar de rețea. Pentru o utilizare simplă, cotidiană, globală, pentru care platforma a fost concepută și mai ales că poate să fie particularizată și continuată de către utilizatori, consider că platforma a atins scopul pentru care a fost creată.



## Capitolul 2. Aplicația "Macroul"

### 2.1. Povestea

Când m-am apucat să lucrez la aplicație, aveam deja o suită de mici funcționalități pe care le-am făcut treptat, începând din liceu cu o bibliotecă utilă pentru diverse programe (printrscreen, click-uri, taste, crearea unei ferestre pe care puteam desena, încărcare de BMP-uri pentru jocuri etc.), care merita să fie continuată cu noile cunoștințe dobândite în facultate dar mai ales la locul de muncă. Acolo a apărut altă librărie de acest tip, cu funcții pentru conectarea în C la o bază de date, conversii între wchar\_t și char, funcții pentru scris în regiștri, job-uri pentru baze de date (operațiile CRUD) care folosea framework-ul de la muncă pentru a se conforma cerințelor de acolo. Acesta a fost de altfel primul loc unde am văzut cât de mult cod pur formal învăluie o idee, când un banal select ajungea să ia două fișiere a câte 100 de rânduri de cod orientat-obiect plus o structură de date diferită. Evident, părându-mi-se inutil am făcut un șablon de structură în care datele se țineau ca șiruri de caractere și puteau fi extrase direct formate în funcție de tip. Apoi am observat cum cei de la muncă petreceau destul de mult timp configurând diverse sisteme conectându-se prin Telnet/SSH la acestea și executând aceleași instrucțiuni scrise pe două foi A4 și m-am hotărât să îi ajut. Astfel am decis că este momentul pentru o altă automatizare și împreună cu un coleg am făcut un utilitar care găsea toate IP-urile din rețea și se conecta Telnet/SSH la acestea și după ce interoga identitatea lor individualiza un script scris generic, încărca niște fișiere prin WinSCP, îl executa și apoi se închidea. Acest utilitar a făcut ca de la un dispozitiv care se configura în 15 minute să ajungem la 200 de dispozitive care se configurau în aproximativ 2 minute, doar cu un simplu dublu-click. Acesta a fost momentul crucial când am văzut puterea rutinelor pe care le făcusem până atunci, nu utilitatea lor pentru mine ci utilitatea generală, nevoia oamenilor de a avea acces la automatizări. De asemenea, merită menționat faptul că cei de la muncă nici nu se gândeau să automatizeze această rutină sau credeau că nici nu se poate.

Am pornit deci la drum cu o zestre bogată de utilitare a căror folosire se făcea în general programatic și problema la care a trebuit să răspund era "Cum pot crea o soluție astfel încât să permit folosirea utilităților și automatizarea lor de către oricine?". Răspunsul a fost unul pe cât de natural, pe atât de convenabil - o aplicație Desktop, folosind framework-urile cu care eram obișnuit de la muncă. Evident, acestea au trebuit completate cu o parte web, cu care nu eram tocmai familiar. Primul lucru pe care l-am făcut a fost să îmi fac o bază de date și să mă gândesc la o schemă utilizabilă. Astfel, am ajuns la concluzia că orice automatizor trebuie să ofere acces la rutinele create, astfel încât să nu depindă de platforma de utilizare. Am ajuns la un sistem de conturi unde aplicația putea fi totuși folosită și exclusiv offline, unde fiecare automatizare se dispunea sub formă de script ale cărui instrucțiuni bazale aveau o formă foarte simplă, ușor codificabilă.

Trebuia să existe o interfață de modificare a acestor scripturi, pe care le-am numit "Macrouri", datorită similitudinii cu posibilitățile oferite de către programe ca AutoHotKey sau Easy Macro Recorder. Am căutat atunci diverse programe care făceau mai mult sau mai puțin același lucru și m-am întrebat dacă programul meu chiar aduce un plus de valoare sau utilitate unei piețe despre care aflasem că este deja saturată. Dar oare era ea saturată? Pe de o parte fiecare tastatură modernă vine cu programul propriu de macrouri, deci fiecare are la dispoziție

măcar o astfel de aplicație pe lângă volumul impresionant de aplicații de acest tip de pe internet, dar eu nu aveam o astfel de tastatură nouă, nu știam de altul în afară de AutoHotKey și toate pe care le-am găsit erau fie prea complicate, suficient de complicate încât scrierea în C sau Batch devenea preferabilă (AutoHotKey) fie erau extrem de simpliste și inexpressive și nu ofereau puterea de a face nimic în afara exemplelor date, care deja atingeau limitele aplicației astfel încât uneori nici măcar acelea nu mergeau.

Un exemplu ilustrativ este cel al unei aplicații care pretindea că are comandă vocală și se lăuda că la interceptarea mesajului "Read me" executa o comandă. Oricât am încercat și oricui i-am dat executabilul, mesajul înțeles nu a fost niciodată "Read me", mai mult, interpretorul nu era nici măcar determinist. Punând același mesaj înregistrat, de două ori am obținut rezultate diferite, evident niciunul bun. Am ajuns la concluzia că saturația era doar aparentă și că o nișă destul de neexplorată era aceea a programelor de macroui cu comandă vocală internaționalizată. Astfel, am ajuns la ideea activatorilor, dintre care unul este interceptarea unui mesaj dat, în limba română. Întrucât șablonul cu care compar nu este un eșantion vocal, ci un text, verificarea nu depinde de similaritatea vocii mele cu a utilizatorului ci doar de capacitatea lui de a vorbi astfel încât să fie înțeles de Google Translate.

Am încercat în primă fază să îmi fac translatorul meu, dar rezultatele nu au fost satisfăcătoare și m-am consolatat cu gândul că acele mii de oameni în a căror posesie stă o rețea de calcul paralel imensă, care fac cercetare pe seturi uriașe de date, cu acces la un potențial computațional și intelectual aproape nemărginit sunt mult mai apti să facă o astfel de unealtă decât mine iar eu, un om al cărui principal avantaj e o idee, o nuanță, o nișă, sunt mult mai bine pregătit să îmi dezvolt efectiv ideea și nu să reinventez roata rezolvând o problemă care nu ține de specificitățile problemei de la care am pornit.

## 2.2. Activatori

Să dăm însă întâi un exemplu de macrou, pentru ca cititorul să poată lesne urmări continuarea: am vrea să putem naviga printr-un powerpoint/pdf. Mai precis, am vrea ca la un anumit stimul să poată apăsa sus/jos/dreapta/stânga etc.

Un astfel de macrou ar putea fi util în diverse contexte - când utilizatorul nu are un pointer sau când nu dorește achiziționarea unuiia sau când îl uită sau rămâne fără baterie sau pentru toate celelalte cazuri când e nevoie de aceste taste în absența unui proiector precum jucarea unui joc de tetris pe televizor/monitor de la o anumită distanță.

Am putea concepe o serie de activatori utili:

**2.2.1 Comandă vocală:** Util în special pentru cei cu dizabilități (care din nefericire nu pot folosi celelalte activatoare) dar nu numai.

**2.2.2 Apăsarea unei combinații de taste:** De exemplu "Ctrl + Alt + Shift + Double Click Right". Acesta este de departe cel mai popular mod de a activa un macrou în celelalte programe de acest tip. Pentru a continua exemplul, tastele F1 - F12 ar putea duce un powerpoint la slide-urile cheie ale unei prezentări voluminoase, la care s-ar putea să se dorească revenirea rapidă.

**2.2.3 Activarea macroului la un moment de timp ales:** Cum ar fi cazul unei prezentări automate, fără nevoia unui input în timpul prezentării, care să permită deci celui care prezintă să

scrie la tablă în paralel sau să prezinte din spatele amfiteatrului pentru ca cei din față să vadă textul iar cei din spate, care nu vor vedea așa bine să-i audă vocea.

**2.2.4 Introducerea unui card RFID:** Util mai ales în locurile de muncă în care fiecare angajat, având de cele mai multe ori deja un astfel de card (fie de la RATB sau chiar de acces în clădire) poate să treacă respectivul card prin fața unui cititor și să se execute un anumit script.

**2.2.5 Activarea de pe o pagină web:** Permite transformarea unui dispozitiv pe care îl are toată lumea - smartphone-ul într-o telecomandă complet customizabilă pentru desktop. Scenariul cu pointerul este poate cel mai relevant, pentru că telefonul nu are nevoie de baterii AAA pe care nu le poți încărca în timpul prezentării și pointer-ul mai mult ca sigur nu poate rula un script în MATLAB. Acest exemplu a fost principalul motiv pentru introducerea meniurilor care vor fi discutate mai jos.

Astfel, pentru a rezolva problema, un utilizator poate face 4 macroui cu nume sugestive ("Sus", "Stanga", "Dreapta", "Jos") fiecare având codul macrou respectiv ("Taste(O tasta,Ambele,UP)", "Taste(O tasta,Ambele,LEFT)", "Taste(O tasta,Ambele,RIGHT)", "Taste(O tasta,Ambele,DOWN)") și le poate pune într-un meniu care arată ca în **Figura 2.1.** și să obțină aceeași funcționalitate ca cea a unui pointer folosind propriul telefon.

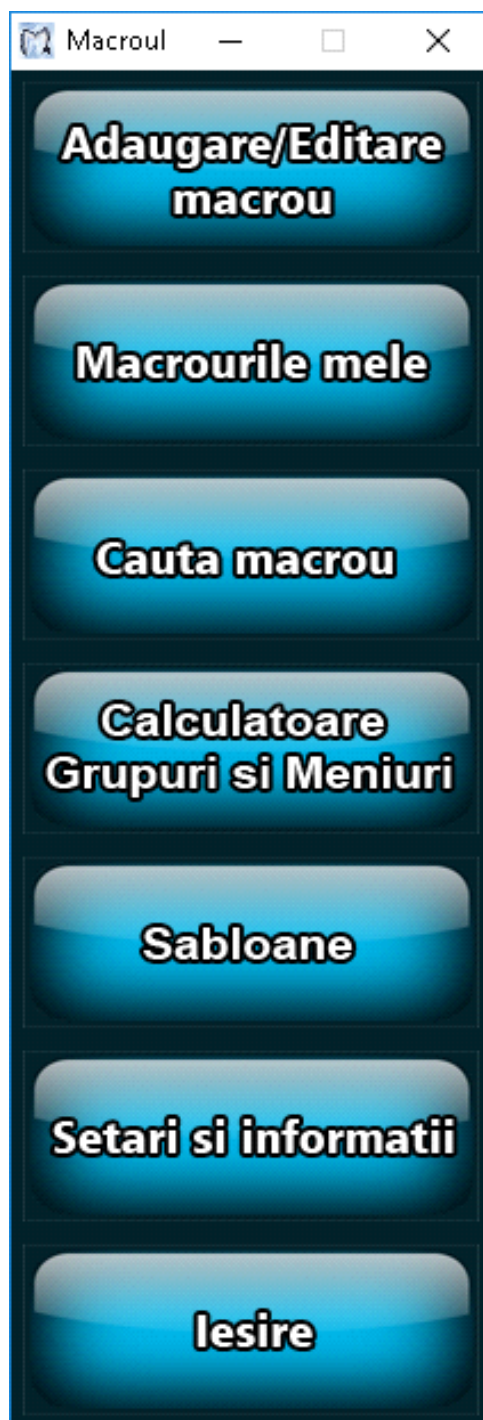


**Figura 2.1.** Exemplu de meniu

## 2.3. Modul de operare

Când este deschisă aplicația, utilizatorul este întâmpinat de fereastra de login unde poate să își creeze un cont nou sau să intre în cel existent din aceeași fereastră. Nu există restricții asupra parolei dar este recomandată o parolă destul de puternică din moment ce poate să ofere acces la calculatorul personal. Când se crează un cont nou, se trimite pe acea adresă de email un cod care trebuie introdus în aplicație pentru validarea adresei.

Meniul principal (**Figura 2.2.**) este format din 7 butoane ("Adaugare/Editare macrou", "Macrourele mele", "Cauta macrou", "Calculatoare Grupuri si Meniuri", "Sabloane", "Setari si informatii", "Iesire"). Detaliem acum funcționalitatea acestora:



**Figura 2.2.** Meniul principal

### **2.3.1. "Adaugare/Editare macrou"**

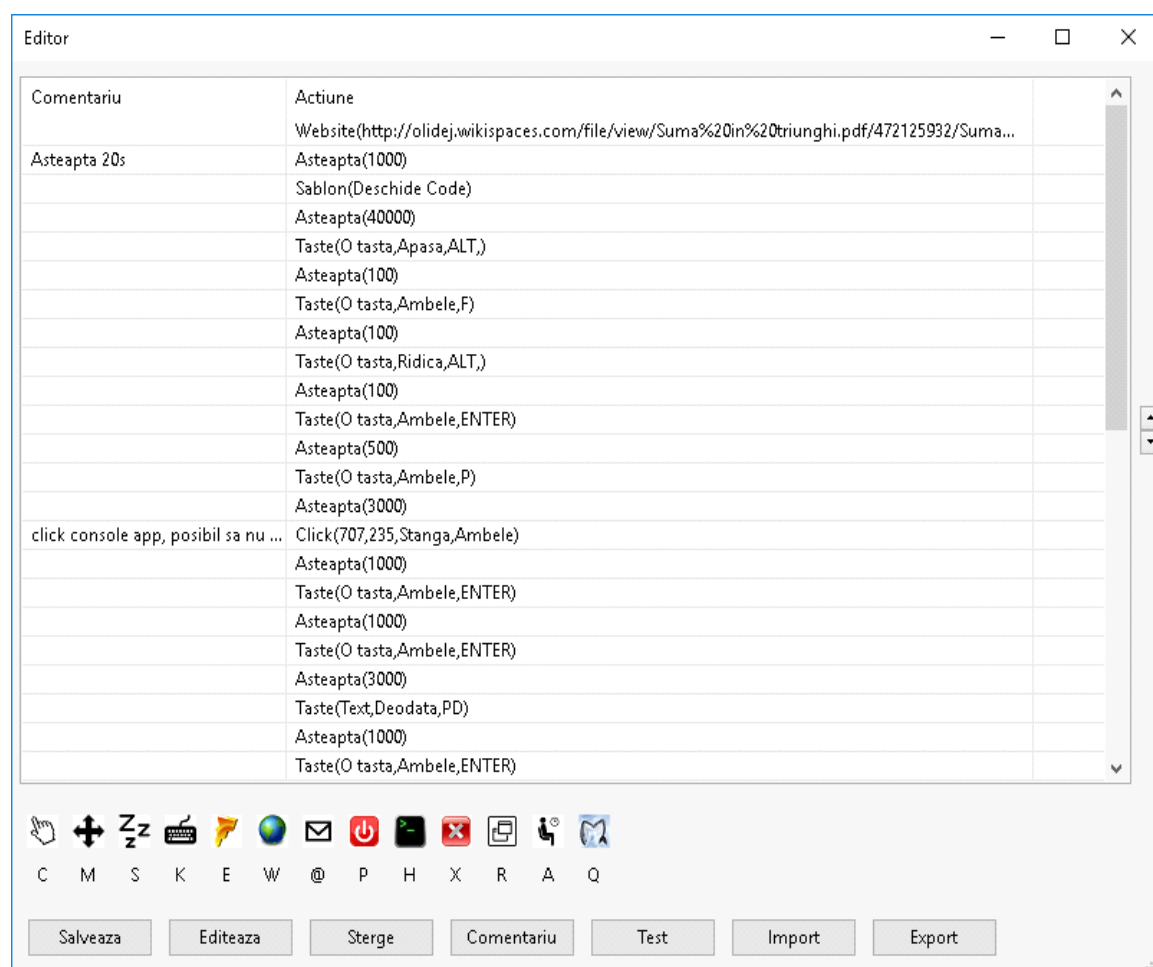
În acest panou, utilizatorul vede o listă cu toate macrourele care îi aparțin (ID-ul lor, numele, o scurtă descriere și nivelul de vizibilitate de către alți utilizatori - public sau privat) și poate să editeze sau să șteargă macrourele existente sau să adauge un macrou nou folosind butoanele de jos. Apăsând pe butonul de editare sau pe cel de adăugare se ajunge la editorul de macrouri, cel mai important și cel mai folosit panou din această aplicație (**Figura 2.4.**).

ID	Nume	Descriere	Public	
42	Mona Lisa	Deseneaza Mona Lisa	Da	
47	Luceafarul	Scrie luceafarul in Notepad	Da	
48	W		Da	
49	S		Da	
50	A		Da	
51	D		Da	
52	Stanga		Da	
53	Dreapta		Da	
54	Sus		Da	
55	Jos		Da	
56	Inchide PC	Inchide calculatorul in 20 de minute	Da	
58	Deschide My Computer		Da	
59	espeek	Spune prima strofa din Luceafarul	Da	
62	Problema triumphiului		Da	

**Figura 2.3.** Meniul "Adaugare/Editare macrou"

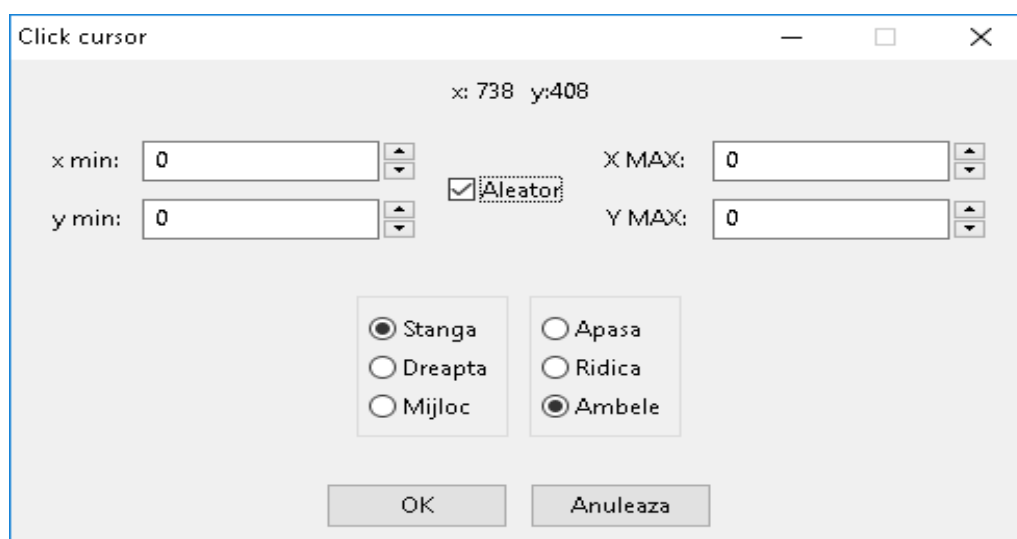
### 2.3.2. Editorul de macrouri

Butoanele de jos sunt destul de evidente ("Salveaza" - salvează macroul și toate schimbările făcute, "Editeaza" - editează rândul selectat, "Sterge" - șterge rândul selectat, "Comentariu" - adaugă sau editează un comentariu existent pe rândul respectiv - de exemplu am folosit pe al doilea rând un comentariu care să îmi amintească faptul că trebuie modificat la sfârșit durata acelei instrucțiuni, iar pe rândul 15 am menționat faptul că acea poziție la care se face click se poate schimba în funcție de calculator și trebuie verificată, "Test" - execută instrucțiunea de pe rândul selectat, iar în cazul în care nu este nimic selectat, execută toate rândurile, "Import/Export" - încarcă/salvează fișiere ".Macrou" care se pot deschide ulterior ca niște fișiere executabile) dar în schimb cele de deasupra lor ne arată toate instrucțiunile simple de care dispunem.



**Figura 2.4.** Editorul

### 2.3.3. C (Click)



**Figura 2.5.** Dialogul "Click"

În acest dialog, utilizatorul poate să aleagă dacă poziția cursorului în momentul click-ului să fie aleatoare sau nu (această opțiune este utilă în special celor care au nevoie de acest "random" - cei care vor să testeze o aplicație și trebuie să încerce cât mai multe butoane în ordine aleatoare - cât și pentru a ascunde faptul că folosesc un script în condițiile în care sunt înregistrate pozițiile cursorului în cazul jocurilor). Astfel, se poate crea un "Bounding box" având vârfurile sus-stânga și jos-dreapta date de (x min, y min) și (X MAX, Y MAX). Evident că era nevoie să existe opțiunea de a alege între cele mai folosite butoane ale mouse-ului (Stânga, Dreapta, Mijloc) dar și tipul de eveniment care se realizează (dacă este de apăsare, ridicare sau întâi apăsare și apoi ridicare - echivalentul unui click normal), deoarece sunt cazuri în care ne este util așa ceva (drag and drop). De asemenea, textul de sus ("x:738 y:408") se schimbă o dată cu mișcarea mouse-ului de către utilizator pentru ca acesta să cunoască poziția cursorului și implicit ce date trebuie să introducă.

### Sintaxa:

- Pentru click-uri aleatoare:

Click(x min|y min,X MAX|Y MAX,Stanga/Dreapta/Mijloc,Apasa/Ridica/Ambele)

Exemplu: Click(0|100,0|100,Stanga,Ambele)

- Pentru restul:

Click(x,y,Stanga/Dreapta/Mijloc,Apasa/Ridica/Ambele)

Exemplu: Click(7,7,Dreapta,Ridica)

### Detalii implementare:

Pentru a obține poziția cursorului se folosesc instrucțiunile "POINT p; GetCursorPos(&p);" iar rezultatul o să se găsească în "p.x" și "p.y" . Pentru random am folosit generatorul pseudo-aleator Mersenne Twister 19937 deoarece oferă un aleatorism "mai bun" decât cel standard oferit de către limbajul C (rand). Pentru a executa o comandă de Click obținem cele 4 elemente separate prin virgulă aplicând un String Tokenizer, verificăm dacă primul element conține sau nu caracterul '|' iar în caz afirmativ, împărțim iarăși în 2 tokeni care reprezintă limitele pentru axa OX și alegem x corespunzător, analog pentru al 2-lea element (y). Având toate informațiile necesare, punem cursorul la coordonatele (x,y) folosind comanda "SetCursorPos(x,y);" și în funcție de restul argumentelor avem 9 cazuri după cum urmează:

```
if(a1=="Stanga")
{
    if(a2=="Apasa")
        mouse_event(MOUSEEVENTF_LEFTDOWN, 0, 0,0,0);
    if(a2=="Ridica")
        mouse_event(MOUSEEVENTF_LEFTUP,0,0,0,0);
    if(a2=="Ambele")
        mouse_event(MOUSEEVENTF_LEFTDOWN | MOUSEEVENTF_LEFTUP, 0, 0,0,0);
}
if(a1=="Dreapta")
{
    if(a2=="Apasa")
        mouse_event(MOUSEEVENTF_RIGHTDOWN, 0, 0,0,0);
    if(a2=="Ridica")
        mouse_event(MOUSEEVENTF_RIGHTUP,0,0,0,0);
    if(a2=="Ambele")
        mouse_event(MOUSEEVENTF_RIGHTDOWN | MOUSEEVENTF_RIGHTUP, 0, 0,0,0);
}
if(a1=="Mijloc")
```

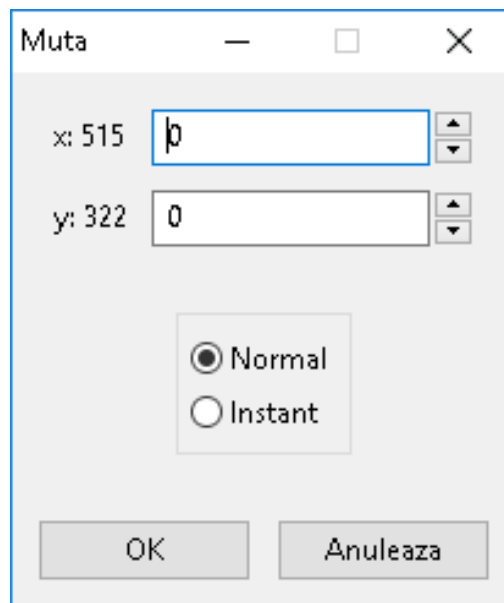
```

{
  if(a2=="Apasa")
    mouse_event(MOUSEEVENTF_MIDDLEDOWN, 0, 0,0,0);
  if(a2=="Ridica")
    mouse_event(MOUSEEVENTF_MIDDLEUP,0,0,0,0);
  if(a2=="Ambele")
    mouse_event(MOUSEEVENTF_MIDDLEDOWN|MOUSEEVENTF_MIDDLEUP, 0, 0,0,0);
}

```

Consider aceste detalii importante pentru cei care doresc să modifice programul dar și pentru cei care doresc să folosească una dintre aceste comenzi în programele lor sau doar sunt curioși de metoda folosită. O să mențin astfel aceeași structură pe parcursul lucrării.

### 2.3.4. M (Move / Mută)



**Figura 2.6.** Dialogul "Muta"

În acest dialog, utilizatorul poate să aleagă poziția la care o să ajungă cursorul după executarea acestei instrucțiuni. Se poate folosi împreună cu alte instrucțiuni, de exemplu Click pentru a obține efectul de "drag and drop". La fel ca la Click, se cunoaște în permanență poziția curentă pentru a ajuta la introducerea datelor. Utilizatorul poate alege modul în care cursorul o să meargă pe ecran, acesta poate să fie instant sau să imite mișcarea unui om, introducând aleatorism.

#### **Sintaxa:**

-Muta(x,y,Normal/Instant)

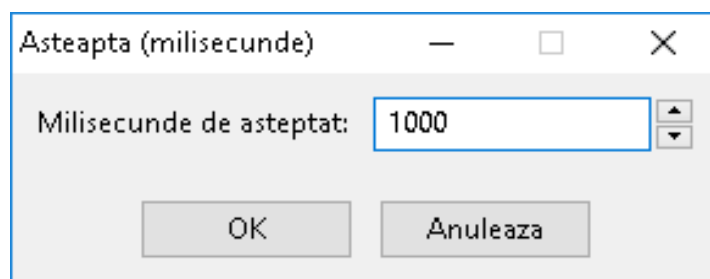
Exemplu: Muta(420,66,Normal)

#### **Detalii implementare:**

Se folosește direct "SetCursorPos(x,y);" pentru cazul instant sau o funcție care are Z% șanse să meargă în direcția bună după care așteaptă un număr random de milisecunde și repetă până când ajunge în punctul dorit (dat ca parametru).



### 2.3.5. S (Sleep / Așteaptă)



**Figura 2.7.** Dialogul "Asteapta"

Deoarece unele comenzi pot să fie prea rapide, este nevoie să introducem un timp de așteptare între acestea. Pentru a nu pune peste tot această așteptare, în setări există un câmp în care putem seta câte milisecunde să aștepte programul între oricare două instrucțiuni succesive. Această setare este utilă în special pentru calculatoare mai vechi care au nevoie de această perioadă de timp în plus peste tot.

#### **Sintaxa:**

-Asteapta(Milisecunde)

Exemplu: Asteapta(1000)

#### **Detalii implementare:**

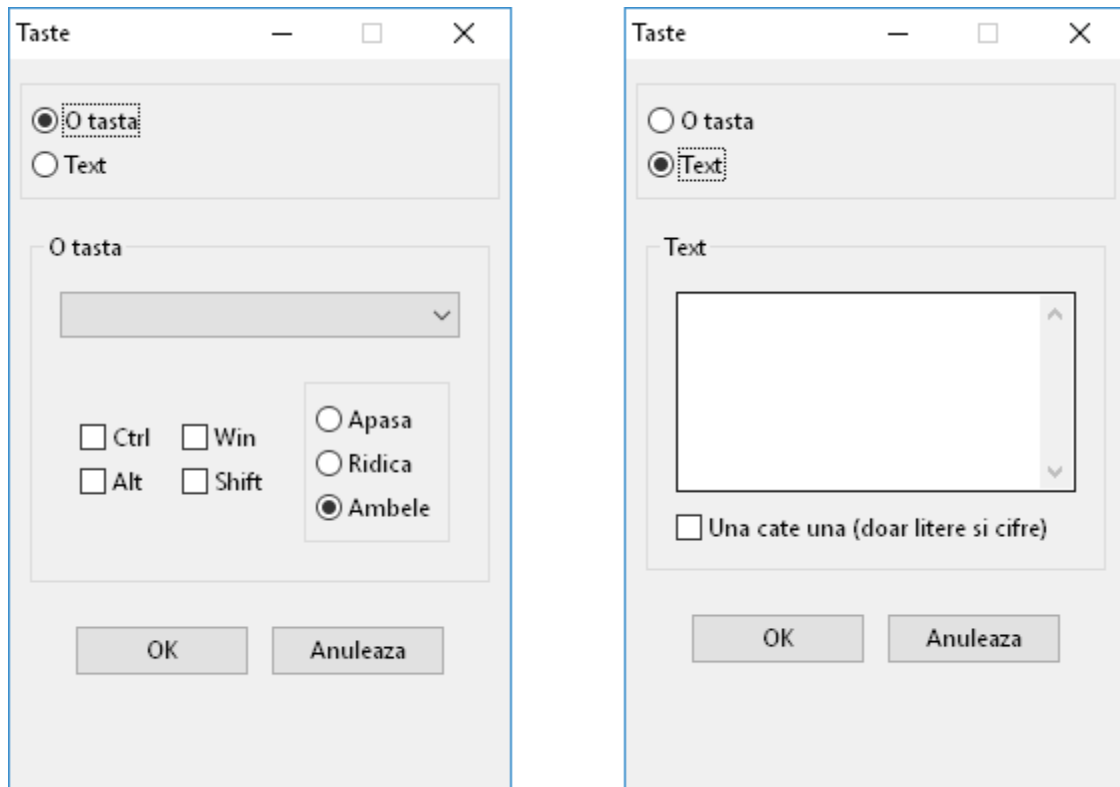
Am folosit comanda "Sleep(x);" din windows.h. Pentru cei ce doresc să implementeze în Linux, pot folosi "sleep(unsigned int seconds);" (pentru secunde) dinunistd.h sau "usleep(useconds\_t usec)" (pentru microsecunde).

### 2.3.6. K (Keyboard / Taste)

Pentru partea de tastatură utilizatorul are 2 variante: să trimită o combinație de taste (Ctrl + Alt + Shift + F) sau să introducă un text care o să fie scris.

**O tastă:** Analog cu sistemul de la Click, se poate alege tipul de eveniment asociat tastei ce urmează să fie simulată (Apasă / Ridică / Ambele). Tastele speciale (Ctrl, Alt, Win, Shift) se pot folosi independent (util în special pentru Win, când dorim să deschidem meniul windows-ului) lăsând liber în choice box (**Figura 2.8.** stânga). Dacă se dorește totuși folosirea mai multor taste speciale ordinea în care ele sunt apăsată este: Ctrl, Win, Alt, Shift, Tastă selectată.

**Text:** Se introduce în căsuța pentru text datele ce urmează a fi scrise în momentul rulării. Este important de menționat faptul că în cazul în care nu este selectată opțiunea "Una câte una" textul este scris folosind clipboard-ul (echivalentul lui Ctrl + V). Deci textul este scris instant și tot ce era salvat înainte în clipboard (echivalentul lui Ctrl + C) este șters / pierdut. Dacă nu se dorește acest lucru, există posibilitatea de a simula pe rând fiecare caracter din text folosind opțiunea menționată mai sus cu anumite restricții. Mi s-a părut important ca la această instrucțiune să se poată introduce și diacritice și am urmărit ca această aplicație să poată funcționa în orice limbă, chiar dacă a adus un plus de muncă în partea de implementare.



**Figura 2.8.** Dialogul " Taste "

### Sintaxa:

-Pentru "O tastă":

Taste(O tasta,Apasa/Ridica/Ambele,CTRL,WIN,ALT,SHIFT,Tastă)  
(Cu CTRL, WIN, ALT și SHIFT opționale)

Exemplu: Taste(O tasta,Ambele,ENTER) și Taste(O tasta,Ambele,WIN,)

-Pentru "Text":

Taste(Text,Deodata/Intre rupt, Text)

Exemplu: Taste(Text,Deodata,Exemplu)

### Detalii implementare:

Pentru cazul "O Tastă" am folosit "keybd\_event(GetVK(x),0xb8,0,0);" pentru a apăsa și "keybd\_event(GetVK(x),0xb8,KEYEVENTF\_KEYUP,0);" pentru a ridica o tastă unde funcția GetVK(string) primește un string ("DEL", "F7", "SHIFT" etc.) și întoarce o tastă virtuală asociată stringului ( if(x=="ENTER")return VK\_RETURN; ).

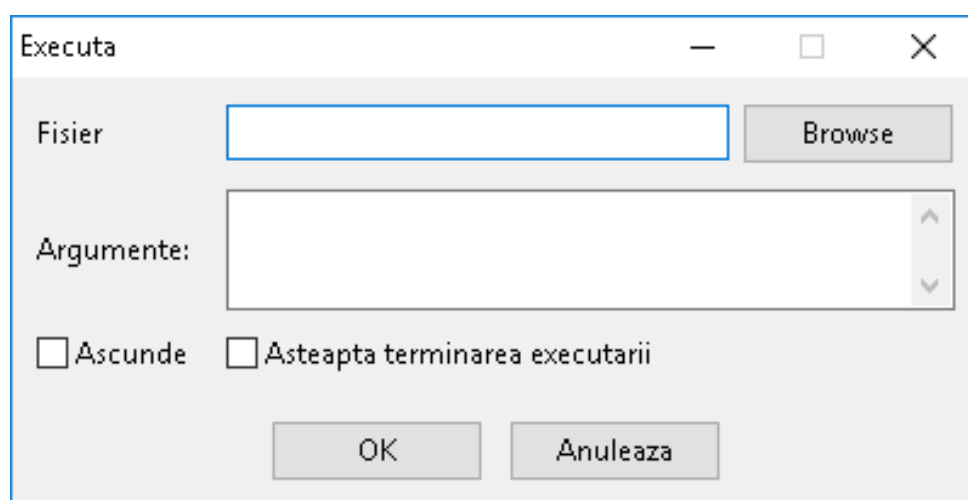
Pentru cazul "Text":

```
const wchar_t* v =temp2;//unde temp2 e stringul pe care vrem să îl scriem
const size_t len = wcslen(v) + 1;
HGLOBAL hMem = GlobalAlloc(GMEM_MOVEABLE, len*sizeof(wchar_t));
memcpy(GlobalLock(hMem), v, len*sizeof(wchar_t));
GlobalUnlock(hMem);
OpenClipboard();
EmptyClipboard();
SetClipboardData(CF_UNICODETEXT, hMem);
CloseClipboard();
keybd_event(0x11, 0, 0, 0); // press ctrl
keybd_event(0x56, 0, 0, 0); // press v
keybd_event(0x56, 0, 2, 0); // release v
keybd_event(0x11, 0, 2, 0); // release ctrl
```

Cum am mai spus, aceste detalii sunt importante și pentru cei care vor să folosească părți din acest program în aplicațiile lor. Un exemplu bun este cel în care se dorește un buton care să copieze în clipboard un text mai lung sau mai complicat de scris (un URL, așa cum vedem în multe aplicații).

Aceste patru instrucțiuni de până acum (Click, Muta, Asteapta, Taste) sunt cele mai utile. Acestea reprezintă principalele metode de input (mouse și tastatură) și sunt esențiale pentru orice program de acest tip.

### 2.3.7. E (Execută)



**Figura 2.9.** Dialogul "Executa"

Această instrucțiune este utilă pentru că permite folosirea sincronizată sau nu a altor aplicații deja existente, executând programe scrise în diverse limbaje de programare. De exemplu m-am găsit de multe ori în situația în care vroiam să execut un fișier în C++ apoi unul în Python deoarece dispunea de anumite librării / implementări mai bune. Un alt motiv pentru introducerea acestei instrucțiuni este acela că uneori este nevoie de rezolvarea unei probleme mai puțin generale care nu se poate rezolva (sau este foarte greu de găsit) folosind instrucțiunile simple oferite de MacroU. Existând numeroase aplicații de tipul "command-line" și deoarece aceste aplicații se pot scrie relativ ușor în principalele limbaje de programare, a fost firească introducerea acestei comenzi.

#### **Sintaxa:**

- Executa(Cale către fișier, Sync/Async, Ascunde/Arata, Argumente)

Exemplu: Executa(C:\Program Files (x86)\eSpeak\command\_line\espeak.exe, Async, Arata, -vro+f3 "Exemplu de text ce urmează să fie rostit de un sintetizator de voce, iar aplicația nu o să aștepte terminarea").

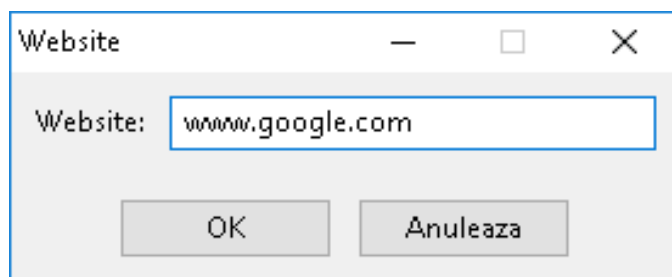
#### **Detalii implementare:**

În mod normal, pentru acest tip de problemă aș folosi "system(string)" care este echivalent cu scrierea în CMD a stringului respectiv, însă o să apară fereastra de CMD, lucru pe

care nu îl dorim mereu. O altă abordare care rezolvă problema ascunderii este "ShellExecute" care oferă prin ultimul argument posibilitatea de a arăta sau nu fereastra pe care o Deschidem(SW\_HIDE) însă nu putem alege dacă așteptăm sau nu terminarea executării programului rulat. Astfel a fost nevoie de ceva mai puternic și anume "ShellExecuteEx".

```
SHELLEXECUTEINFO ShExecInfo = {0};
ShExecInfo.cbSize = sizeof(SHELLEXECUTEINFO);
if(b[0]!='1') //Dacă așteptăm terminarea executării programului
    ShExecInfo.fMask = SEE_MASK_NOCLOSEPROCESS;
ShExecInfo.fMask |= SEE_MASK_NOZONECHECKS;
ShExecInfo.hwnd = NULL;
ShExecInfo.lpVerb = NULL;
ShExecInfo.lpFile = a; //calea către fișier
ShExecInfo.lpParameters = d; //argumentele
ShExecInfo.lpDirectory = NULL;
if(c[0]!='1')// dacă dorim să afișăm sau nu fereastra executabilului
    ShExecInfo.nShow = SW_HIDE;
else
    ShExecInfo.nShow = SW_SHOW;
ShExecInfo.hInstApp = NULL;
ShellExecuteEx(&ShExecInfo);
if(b[0]!='1')//Așteptăm terminarea executării programului
    WaitForSingleObject(ShExecInfo.hProcess,INFINITE);
```

### 2.3.8. W (Website)



**Figura 2.10.** Dialogul "Website"

Este evident că se puteau folosi celelalte instrucțiuni deja menționate (Click și Taste) pentru realizarea acestuia, însă acestea nu sunt tocmai recomandate (mai ales Click-ul) deoarece la cele mai mici schimbări ale rezoluției, ale browser-ului etc. pot aduce probleme. Mai ales în cazul în care se dorește publicarea macroului sau folosirea lui pe mai multe calculatoare. A fost deci nevoie de introducerea acestei instrucțiuni care să meargă pe orice calculator folosind browserul default de pe acea mașină.

#### **Sintaxa:**

- Website(url)

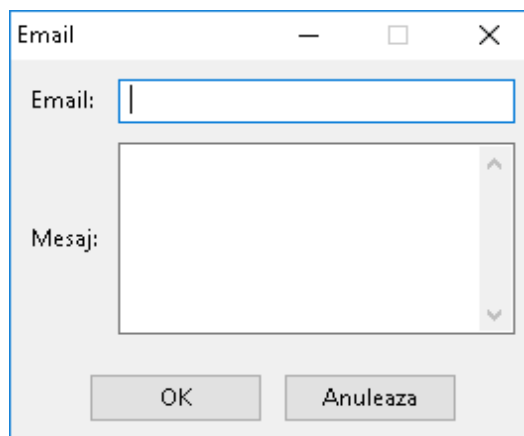
Exemplu: Website([www.google.com](http://www.google.com))

#### **Detalii implementare:**

Surprinzător, funcția ShellExecute menționată mai sus a fost de ajuns.

ShellExecute(NULL, L"open", res, NULL, NULL, SW\_SHOWNORMAL);// unde res este url-ul

### 2.3.9. @ (Email)



**Figura 2.11.** Dialogul "Email"

Folosind executabile care durează mai multe zeci de ore pe mai multe calculatoare, a fost util să aflu când se realiza un eveniment important în loc să verific toate calculatoarele și să caut prin log-uri starea curentă a executabilului. De asemenea, această instrucțiune a fost folosită și la verificarea adreselor de email la înregistrare, dovedind astfel utilitatea ei.

#### **Sintaxa:**

- Email(Adresă email destinație,Mesaj)

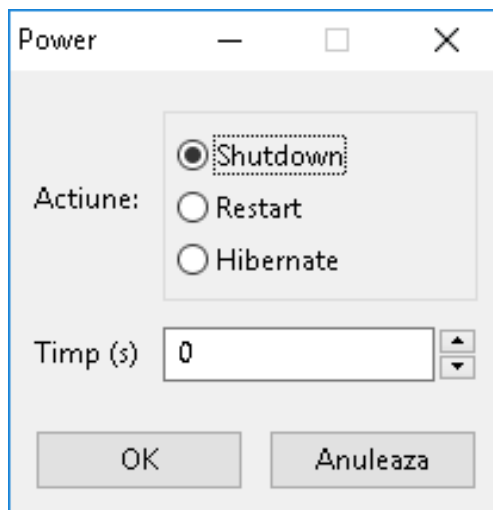
Exemplu: Email([teo2mircea@gmail.com](mailto:teo2mircea@gmail.com),test)

#### **Detalii implementare:**

Cum am mai spus, se găsesc foarte multe executabile tip "command-line". Așa s-a întâmplat și în acest caz. Am folosit un astfel de executabil (mailsend1.19.exe) apelând comanda Executa de mai sus cu argumentele necesare (adresa destinatarului, adresa sursă, parola, subiect, mesaj, să nu apară fereastra CMD etc.). Se poate observa faptul că începe să se vadă utilitatea acestor instrucțiuni încă de la implementare.

### 2.3.10. P (Power)

Am făcut de cel puțin 3 ori o aplicație simplă în C++ în care introduceai un număr reprezentând un număr de secunde după care să se închidă calculatorul. Tot reinstalând sistemul de operare sau din diverse alte motive se pierdeau executabilele respective / codurile sursă. Am folosit de sute de ori aceste mici aplicații și spre surprinderea mea chiar am fost rugat să fac un astfel de timer de către oameni care nu știau că am făcut asta deja. Acestea sunt principalele argumente ce arată utilitatea acestei instrucțiuni.



**Figura 2.12.** Dialogul "Power"

#### **Sintaxa:**

- Power(Shutdown/Restart/Hibernate,Număr secunde)

Exemplu: Power(Shutdown,3600)

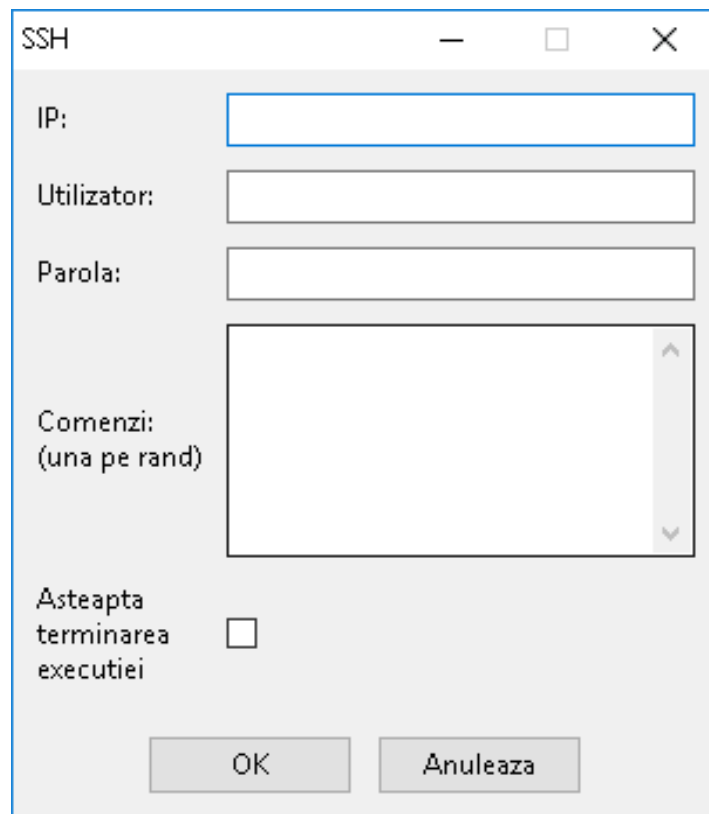
#### **Detalii implementare:**

Am folosit comanda system menționată mai sus, având argumentul "shutdown /s /t secunde /f" de exemplu pentru închidere. Dacă se dorește oprirea acestei comenzi, utilizatorul poate scrie comanda "shutdown -a" în CMD.

```
ichar="";
ichar<<"shut down /";
if(x=="Shut down")
    ichar<<"s /t "<<y;
if(x=="Restart")
    ichar<<"r /t "<<y;
if(x=="Hibernate")
    ichar<<"h";
ichar<<" /f";
system(ichar);
```

### **2.3.11. H (SSH)**

Am urmărit ca funcționalitatea programului să nu se limiteze doar la ce se poate face în interiorul calculatorului. Mai exact, am încercat să fac posibilă crearea unui "Smart house" folosind Macroul. Adăugând diverși activatori (senzori de lumină, temperatură etc.) și având acces în rețea la device-urile necesare, se pot închide luminile (dacă au fost uitate deschise) direct din browser sau să deschidă un ventilator când temperatura trece de un anumit prag.



**Figura 2.13.** Dialogul "SSH"

**Sintaxa:**

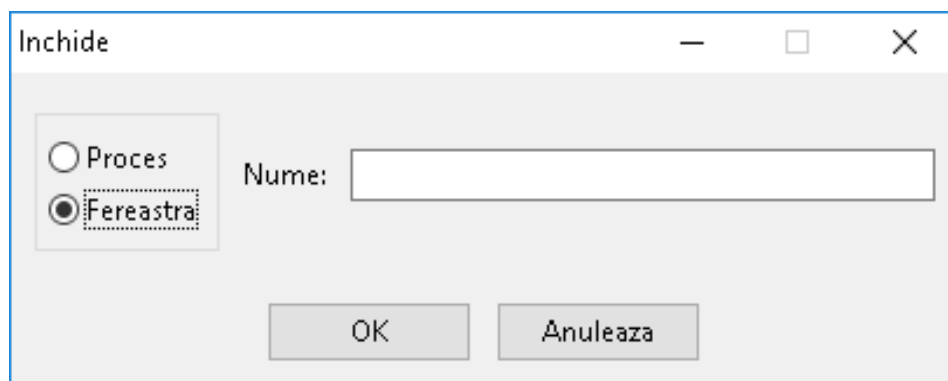
- SSH(Sync/Async,IP,Utilizator,Parolă,Comenzi)

Exemplu: SSH(Sync,192.168.0.103,root,pass,python Desktop/albastru.py)

**Detalii implementare:**

Am folosit putty.exe, un client de SSH cunoscut, cu argumentele necesare (-l pentru nume logare, -pw pentru parolă, -m pentru un fișier cu instrucțiuni) apelând funcția Executa de mai sus.

**2.3.12. X (Închide)**



**Figura 2.13.** Dialogul "Inchide"

Înainte de a folosi multe resurse (sau după ce am terminat de folosit) avem nevoie de a închide procesele sau ferestrele de care nu mai avem nevoie. De exemplu nu vrem ca programul ce o să ruleze zeci de ore care scrie, citește și șterge foarte multe fișiere pe minut să fie încetinit de windows care vrea să indexeze acele fișiere pentru căutări sau să fie scanate de un antivirus când știm că acolo sunt doar txt-uri.

#### **Sintaxa:**

- `Inchide(Proces/Fereastră, Nume proces sau fereastră)`

Exemple: `Inchide(Proces, notepad.exe)`

`Inchide(Fereastră, Untitled - Notepad).`

#### **Detalii implementare:**

Pentru fereastră am căutat HWND-ul ferestrei folosind funcția `FindWindow()` și apoi am trimis un semnal de închidere pentru acea fereastră.

```
HWND x=FindWindow(NULL,window);// unde window e numele introdus de utilizator
PostMessage(x, WM_CLOSE, 0, 0);
```

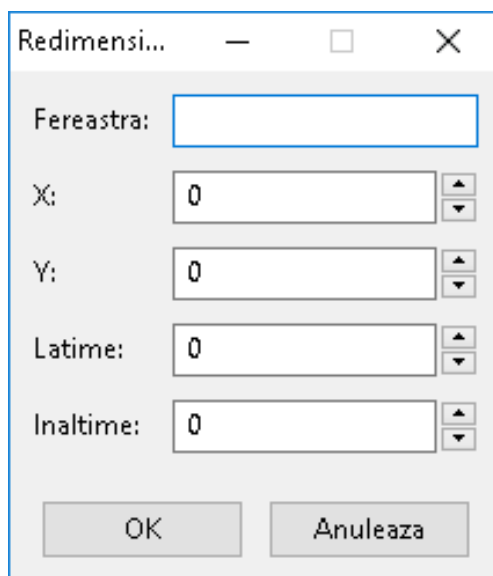
Pentru procese am făcut ceva asemănător, m-am uitat la toate procesele și dacă găseam unul cu același nume cu cel introdus de utilizator, obțin un handle la acel proces și trimit un semnal de terminare (echivalentul lui `kill -9`).

```
void killProcessByName(wchar_t *ProcessName)
{
    HANDLE hSnapShot = CreateToolhelp32Snapshot(TH32CS_SNAPALL, NULL);
    PROCESSENTRY32 pEntry;
    pEntry.dwSize = sizeof(pEntry);
    BOOL hRes = Process32First(hSnapShot, &pEntry);
    while (hRes)
    {
        if (wcscmp(pEntry.szExeFile, ProcessName) == 0)
        {
            HANDLE hProcess = OpenProcess(PROCESS_TERMINATE, 0,
                (DWORD) pEntry.th32ProcessID);
            if (hProcess != NULL)
            {
                TerminateProcess(hProcess, 9);
                CloseHandle(hProcess);
            }
        }
        hRes = Process32Next(hSnapShot, &pEntry);
    }
    CloseHandle(hSnapShot);
}
```

### **2.3.13. R (Resize / Redimensionare)**

Pentru ca un macrou să fie cu adevărat util pentru mai multe persoane, acesta trebuie să funcționeze la fel de bine pe cât mai multe calculatoare (dacă nu chiar pe toate) atunci când este folosit. Un aparent impediment îl aduc numeroasele rezoluții, astfel încât un click la coordonatele (1255,9) ar închide fereastra în care scriu această lucrare, dar nu e sigur că o să fie același efect pentru toți. A fost evidentă atunci nevoia introducerii acestei instrucțiuni pentru generalizare.





**Figura 2.15.** Dialogul "Redimensionare"

**Sintaxa:**

- Redimensionare(x,y,Lățime,Înălțime,Nume fereastră)

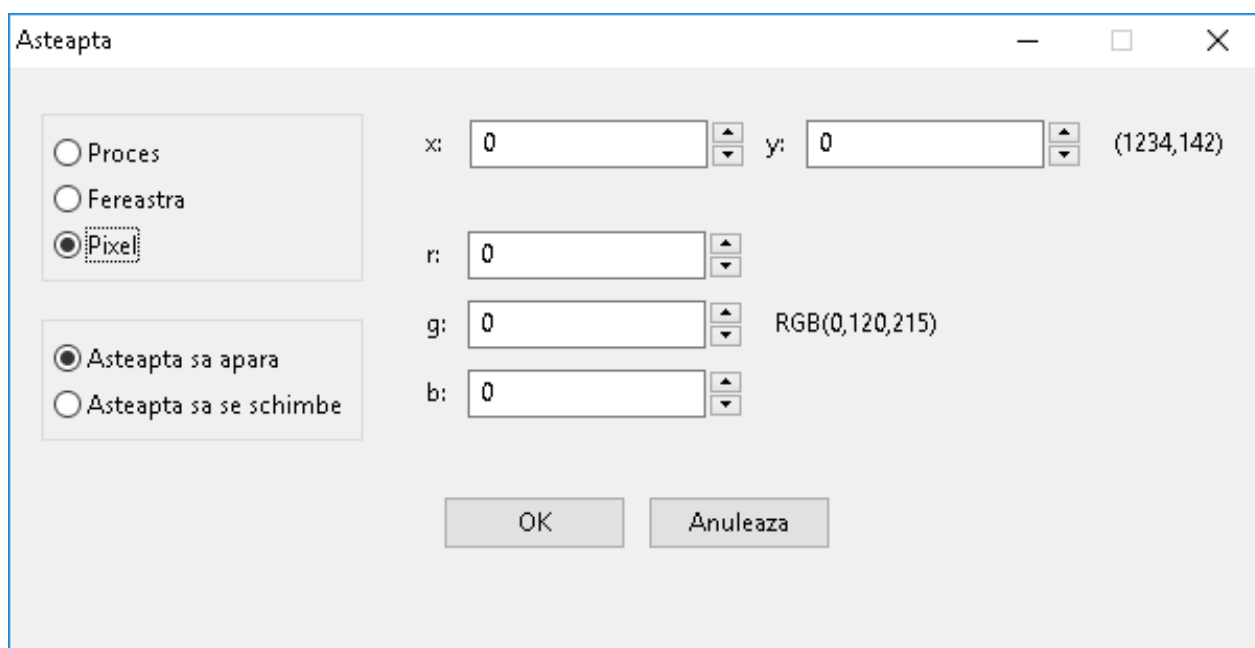
Exemplu: Redimensionare(0,0,1234,500,Untitled - Notepad).

**Detalii implementare:**

Am folosit ca mai sus o funcție din familia FindWindow pentru a găsi fereastra dorită împreună cu SetWindowPos.

`SetWindowPos(FindWindowA(NULL,fereastră),0,x,y,width,height,SWP_SHOWWINDOW|SWP_NOZORDER);`

**2.3.14. A (Așteaptă)**



**Figura 2.16.** Dialogul "Asteapta\_schimbare"

După ce deschidem un proces sau o nouă fereastră, timpul până când acestea apar efectiv diferă foarte mult de la un calculator la altul. O altă problemă apare în momentul în care procesul sau fereastra respectivă nu se pot deschide din cauza unei erori oarecare. Este periculos să continuăm să facem click-uri fără a ne asigura că fereastra este deschisă (sau culoarea pixel-ului nu este cea așteptată). Așadar pentru generalizare și pentru siguranță a fost introdusă această comandă.

### Sintaxa:

Pentru fereastră și proces:

- Asteapta\_schimbare(Proces/Fereastră,Asteapta sa apară/Asteapta sa dispară,Nume)

Exemple: Asteapta\_schimbare(Proces,Asteapta sa apară,notepad.exe)

Asteapta\_schimbare(Fereastră,Asteapta sa dispară,Untitled - Notepad)

Pentru pixeli:

- Asteapta\_schimbare(Pixel,Asteapta sa se schimbe,x,y)

- Asteapta\_schimbare(Pixel,Asteapta sa apară,x,y,R,G,B)

Exemple: Asteapta\_schimbare(Pixel,Asteapta sa se schimbe,985,303)

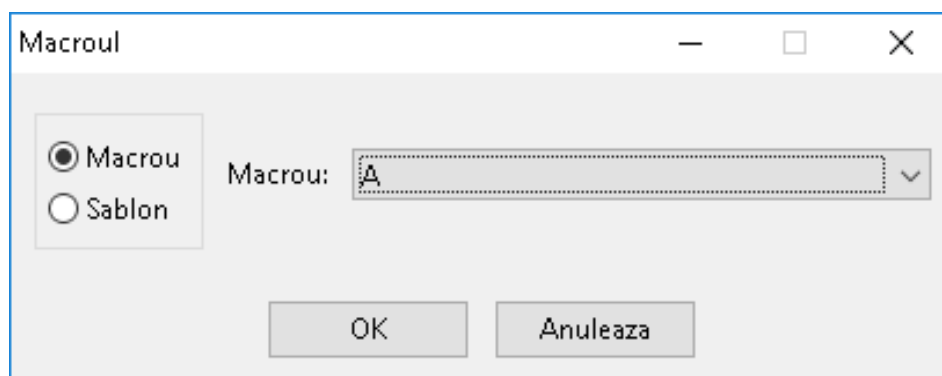
Asteapta\_schimbare(Pixel,Asteapta sa apară,0,0,255,255,0)

### Detalii implementare:

Pentru ferestre și procese este asemănător cu ce a fost deja la instrucțiunea Inchide. Pentru pixeli, verific la fiecare 50 ms dacă a apărut culoarea așteptată (în cazul în care așteptam apariția unei culori anume) sau dacă s-a schimbat culoarea pixel-ului la poziția cerută (când așteptăm o schimbare).

```
if(ch2==0)//sa apară
{
    wxString R,G,B;
    R=tok.GetNextToken();
    G=tok.GetNextToken();
    B=tok.GetNextToken();
    COLORREF color;
    do
    {
        Sleep(50);
        HDC dc = GetDC(NULL);
        color = GetPixel(dc, wxAtoi(x), wxAtoi(y));
        ReleaseDC(NULL, dc);
    }while(wxAtoi(R)!=GetRValue(color) || wxAtoi(G)!=GetGValue(color) || wxAtoi(B)!=GetBValue(color));
}
if(ch2==1)//sa se schimbe
{
    HDC dc = GetDC(NULL);
    COLORREF color = GetPixel(dc, wxAtoi(x), wxAtoi(y));
    ReleaseDC(NULL, dc);
    int r=(int)GetRValue(color),g=(int)GetGValue(color),b=(int)GetBValue(color);
    do
    {
        Sleep(50);
        GetDC(NULL);
        color = GetPixel(dc, wxAtoi(x), wxAtoi(y));
        ReleaseDC(NULL, dc);
    }while(r==(int)GetRValue(color) && g==(int)GetGValue(color) && b==(int)GetBValue(color));
}
```

### 2.3.15. Q (Macrou / Sablon)



**Figura 2.17.** Dialogul "Macrou"

Ultimele două instrucțiuni dau cursivitate și recursivitate macroului. Folosind comenzile Macrou și Sablon putem apela alte macrouri deja definite (sau chiar macroul curent) sau șabloane (definite mai jos). Astfel nu trebuie să scriem de fiecare dată aceleași șiruri lungi de instrucțiuni care fac același lucru. Un exemplu bun este șablonul "Deschide" care apasă butonul start, introduce numele programului care se dorește a fi deschis (windows 7+) și apasă enter:

```
Taste(O tasta,Ambele,WIN,)  
Asteapta(1000)  
Taste(Text,Deodata,Y)  
Asteapta(1000)  
Taste(O tasta,Ambele,ENTER)
```

#### **Sintaxa:**

- Macrou(ID)
- Sablon(Text)

Exemple: Macrou(50)

Sablon(Deschide Paint)

#### **Detalii implementare:**

Pentru Macrou am folosit funcția deja existentă care execută un macrou primind ID-ul acestuia. Am ales să folosesc ID-ul și nu numele deoarece numele poate fi editat și ar putea apărea astfel probleme.

Instrucțiunile simple sunt deci:

- Click
- Muta
- Asteapta
- Taste
- Executa
- Website
- Email
- Power

- SSH
- Inchide
- Redimensionare
- Asteapta\_schimbare
- Macrou
- Sablon

### 2.3.16. "Macrourele mele" (Activatori)

The image shows a Windows-style dialog box titled "Macroul". It contains several sections for configuring macros:

- Voce:** A checkbox followed by a text input field and an "Inregistreaza" button.
- Taste:** A checkbox followed by a text input field and a "Scaneaza" button.
- Program:** A checkbox followed by two date/time pickers (showing "4/23/2017" and "8:20:06 PM") and a button labeled "peste 10 minute".
- Card:** A checkbox followed by a text input field and a "Scaneaza" button.
- Numar de repetitii la activare:** A label followed by a numeric spinner box set to "1".
- Salveaza:** A button at the bottom center.

**Figura 2.18.** Dialogul "Activator"

Macrou ID	Nume	Descriere	Public	Voce	Taste	Program	Card
47	Luceafarul	Scrie luceafarul in Notepad	Da	Luceafărul		24-05-17 21:16:15	854207414
66	da		Da			30-04-17 20:27:00	
69	alarma		Da			14-05-17 15:58:00	
70	f9		Da			22-05-17 22:50:50	
67	for		Da		Pause		
68	sortare		Da		Scroll Lock		
42	Mona Lisa	Deseneaza Mona Lisa	Da				
48	W		Da				
49	S		Da				
50	A		Da				
51	D		Da				
52	Stanga		Da				
53	Dreapta		Da				
54	Sus		Da				
55	Jos		Da				
56	Inchide PC	Inchide calculatorul in 20 de minute	Da				
58	Deschide My Computer		Da				
59	espeek	Spune prima strofa din Luceafarul	Da				
62	Problema triunghiului		Da				

Activator

**Figura 2.19.** Panoul "Macrourele mele"

În acest panou, utilizatorul poate vedea macrourele create și modul lor de activare (Figura 2.19.). După ce selectează un macrou, se pot seta activatorii (Figura 2.18.) bifând checkbox-ul de pe rândul respectiv și apoi:

**Pentru voce**, utilizatorul apasă pe butonul "Inregistreaza", spune cuvintele care o să fie asociate macroului respectiv (de exemplu "test") apoi apasă butonul încă o dată. Pentru siguranță, mesajul este scris și în dialog pe același rând. După ce a fost salvat acest activator, atunci când în microfon este spus cuvântul cheie "macrou" urmat de mesajul salvat ("test") se activează macroul respectiv. Există și pentru șabloane ceva similar (vezi 2.3.19.). În ambele cazuri trebuie menționat faptul că nu se poate folosi un microfon conectat prin USB.

**Detalii implementare:** Cum am mai spus, pentru voce am folosit API-ul de la Google, deoarece este foarte precis și foarte simplu de folosit (au fost necesare 8 rânduri în python):

```
import sys
import speech_recognition as sr
r = sr.Recognizer()
r.pause_threshold=0.5
with sr.Microphone() as source:
    audio = r.listen(source)
try:
    t=r.recognize_google(audio, language = sys.argv[1])
    ...
```

Evident că a fost nevoie să fie transformat într-un program ".exe" pentru a nu fi nevoie ca utilizatorul să instaleze în prealabil Python și trebuiau rezolvate dependențele de librărie etc.

**Pentru taste** este asemănător, se apasă pe butonul "Scaneaza", se introduce o anumită succesiune dorită de taste (și mouse!) și apoi se apasă iar butonul. Din nou, este afișată acea

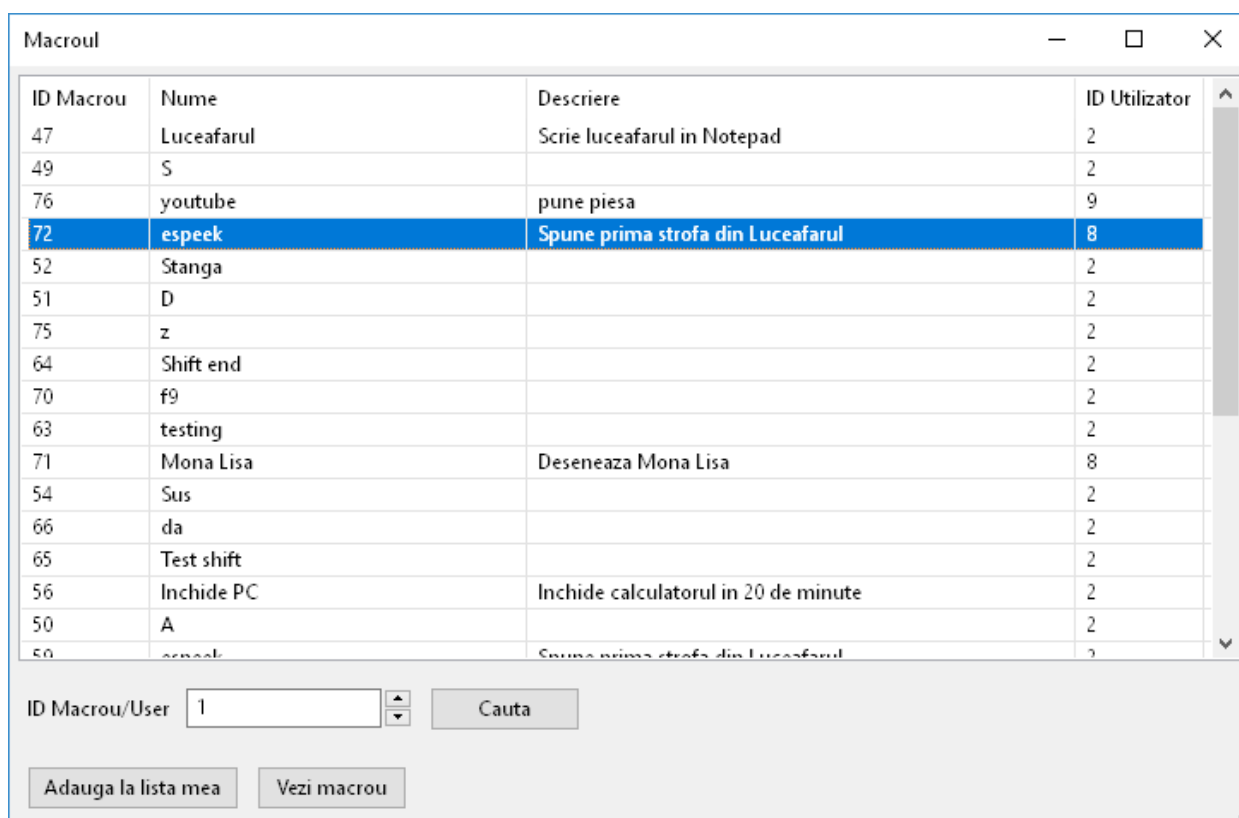
succesiune pentru verificare. După salvare, când sunt folosite acele taste, macroul respectiv se activează.

**Detalii implementare:** Pentru taste și mouse am instalat un hook (pentru fiecare) care este anunțat când este folosit. Alte metode precum "GetAsyncKeyState" ar putea funcționa dar nu le consider la fel de sigure în sensul că se poate produce succesiunea de taste și să nu fie așteptat în acel moment. Un alt argument este acela că în cazul mai multor astfel de activatoare ar fi necesar să verificăm fără oprire fiecare tastă din fiecare combinație salvată ceea ce folosește mai multe resurse și poate să fie nesigură. Folosind hook-urile, verificăm doar atunci când chiar este folosită tastatura sau mouse-ul.

**Pentru program,** folosirea acestuia este foarte simplă, singurul aspect care trebuie menționat este că formatul zilei este același cu formatul windows-ului. Deci, în funcție de calculator poate să difere formatul.

**Pentru card** am folosit un cititor pe care deja îl aveam. Programul funcționează doar cu acest model specific, însă acest activator a fost gândit ca un exemplu că se pot folosi și alte dispozitive mai puțin folosite drept activatori (senzori de temperatură, lumină, mișcare etc.).

### 2.3.17. "Cauta macrou"



The screenshot shows a window titled "Macroul" with a table of macros and a search interface at the bottom. The table has four columns: ID Macrou, Nume, Descriere, and ID Utilizator. The row with ID 72 and name "espeak" is highlighted in blue. Below the table, there is a search bar with the text "ID Macrou/User" and a dropdown menu showing "1". To the right of the search bar is a button labeled "Cauta". Below the search bar are two buttons: "Adauga la lista mea" and "Vezi macrou".

ID Macrou	Nume	Descriere	ID Utilizator
47	Luceafarul	Scrie Luceafarul in Notepad	2
49	S		2
76	youtube	pune piesa	9
72	espeak	Spune prima strofa din Luceafarul	8
52	Stanga		2
51	D		2
75	z		2
64	Shift end		2
70	f9		2
63	testing		2
71	Mona Lisa	Deseneaza Mona Lisa	8
54	Sus		2
66	da		2
65	Test shift		2
56	Inchide PC	Inchide calculatorul in 20 de minute	2
50	A		2
50	espeak	Spune prima strofa din Luceafarul	2

**Figura 2.20.** Panoul "Cauta macrou"

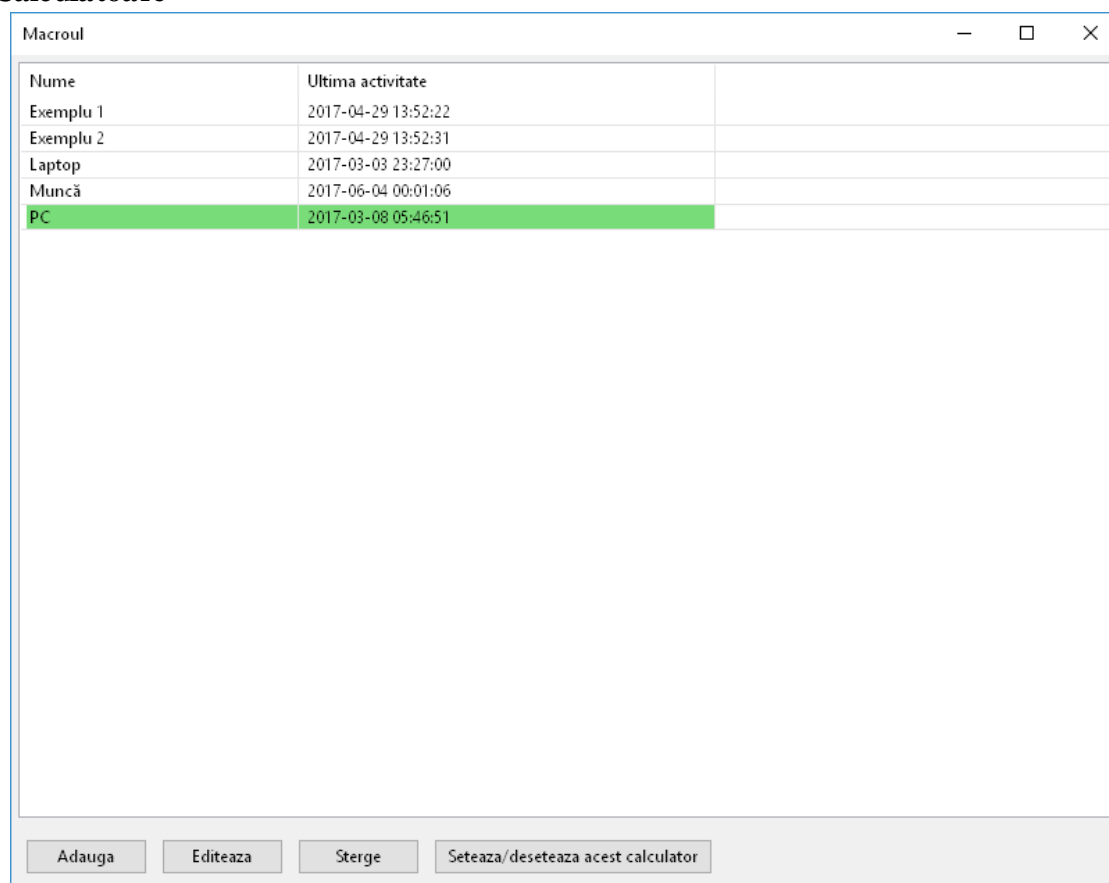
În acest panou, utilizatorul poate vedea toate macrourele făcute publice de către alți utilizatori într-o ordine aleatoare. Acesta poate filtra după ID-ul macroului (dacă îi spune cineva

care este, se poate vedea de exemplu în "Macrourele mele" sau în "Adaugare/Editare macrou") sau după ID-ul utilizatorului care a creat macroul respectiv (se poate vedea în "Setari si informatii"). Butonul "Vezi macrou" deschide un dialog în care utilizatorul poate vedea din ce este format macroul respectiv și implicit să verifice dacă este ceea ce dorea. Butonul "Adauga la lista mea" face o copie identică a macroului respectiv la momentul respectiv și o adaugă la lista utilizatorului. Ulterioarele modificări aduse macroului "sursă" nu modifică macroul copie.

### 2.3.18. "Calculatoare Grupuri si Meniuri"

Ațiunile din aceste trei panouri sunt utile doar pentru website.

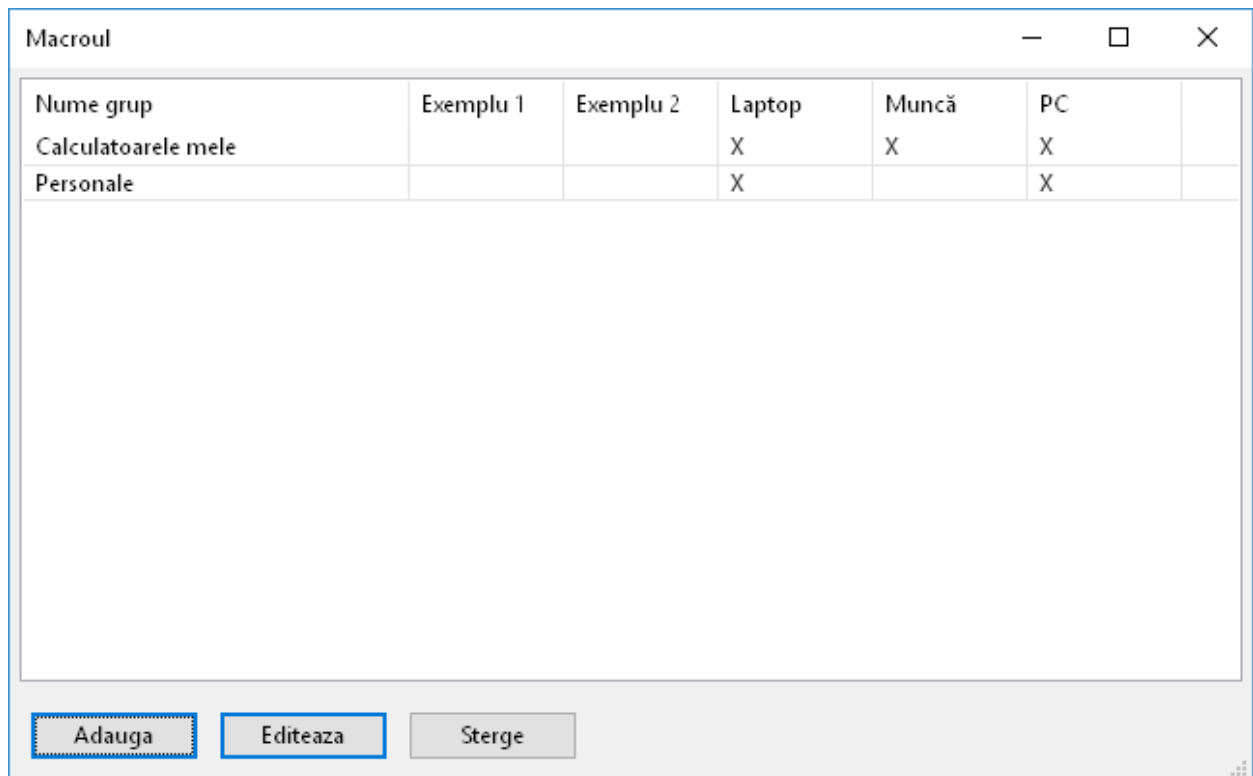
- **Calculatoare**



**Figura 2.21.** Panoul "Calculatoare"

Dacă se dorește comunicarea din afară (de pe website) cu unul dintre calculatoarele pe care este instalat macroul, este necesar ca acestea să fie identificabile. Astfel, fiecare utilizator poate să asocieze un nume cu fiecare calculator. Acest lucru se face foarte ușor din panoul "Calculatoare" (**Figura 2.21.**) unde se pot adăuga, șterge, edita, activa sau dezactiva calculatoare. Pentru cei ce nu vor să fie accesate calculatoarele din afară, aceștia nu trebuie să facă nimic deoarece dacă nu are un nume, calculatorul nu poate să fie accesat.

- **Grupuri**



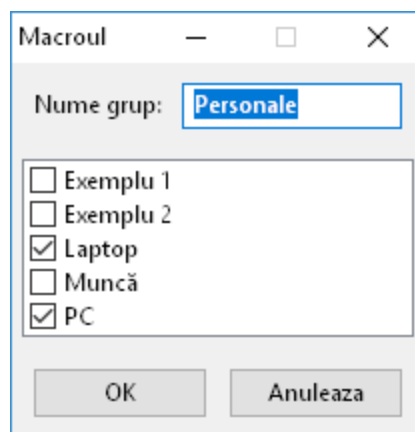
The screenshot shows a window titled 'Macroul' with a table and three buttons at the bottom. The table has columns for 'Nume grup', 'Exemplu 1', 'Exemplu 2', 'Laptop', 'Muncă', and 'PC'. The first row is 'Calculatoarele mele' and the second is 'Personale'. The 'Laptop' and 'PC' columns have 'X' marks in the first two rows. The 'Muncă' column has an 'X' in the first row. The buttons at the bottom are 'Adauga', 'Editeaza', and 'Sterge'.

Nume grup	Exemplu 1	Exemplu 2	Laptop	Muncă	PC
Calculatoarele mele			X	X	X
Personale			X		X

Buttons: Adauga, Editeaza, Sterge

**Figura 2.21.** Panoul "Grupuri"

Uneori, nu este suficient să avem doar o listă cu toate calculatoarele, este posibil să vrem să le și grupăm (de exemplu în timpul unui test de laborator pe mai multe numere). Chiar dacă această grupare se poate face de pe website (există un checkbox în dreptul fiecărui calculator), dacă sunt prea multe calculatoare devine greu de urmărit. Editarea unui grup este simplă trebuie doar selectate calculatoarele dorite ca în **Figura 2.22**.



The screenshot shows a dialog box titled 'Macroul' with a text field for 'Nume grup' containing 'Personale'. Below it is a list of items with checkboxes: 'Exemplu 1', 'Exemplu 2', 'Laptop', 'Muncă', and 'PC'. The 'Laptop' and 'PC' checkboxes are checked. At the bottom are 'OK' and 'Anuleaza' buttons.

Nume grup:

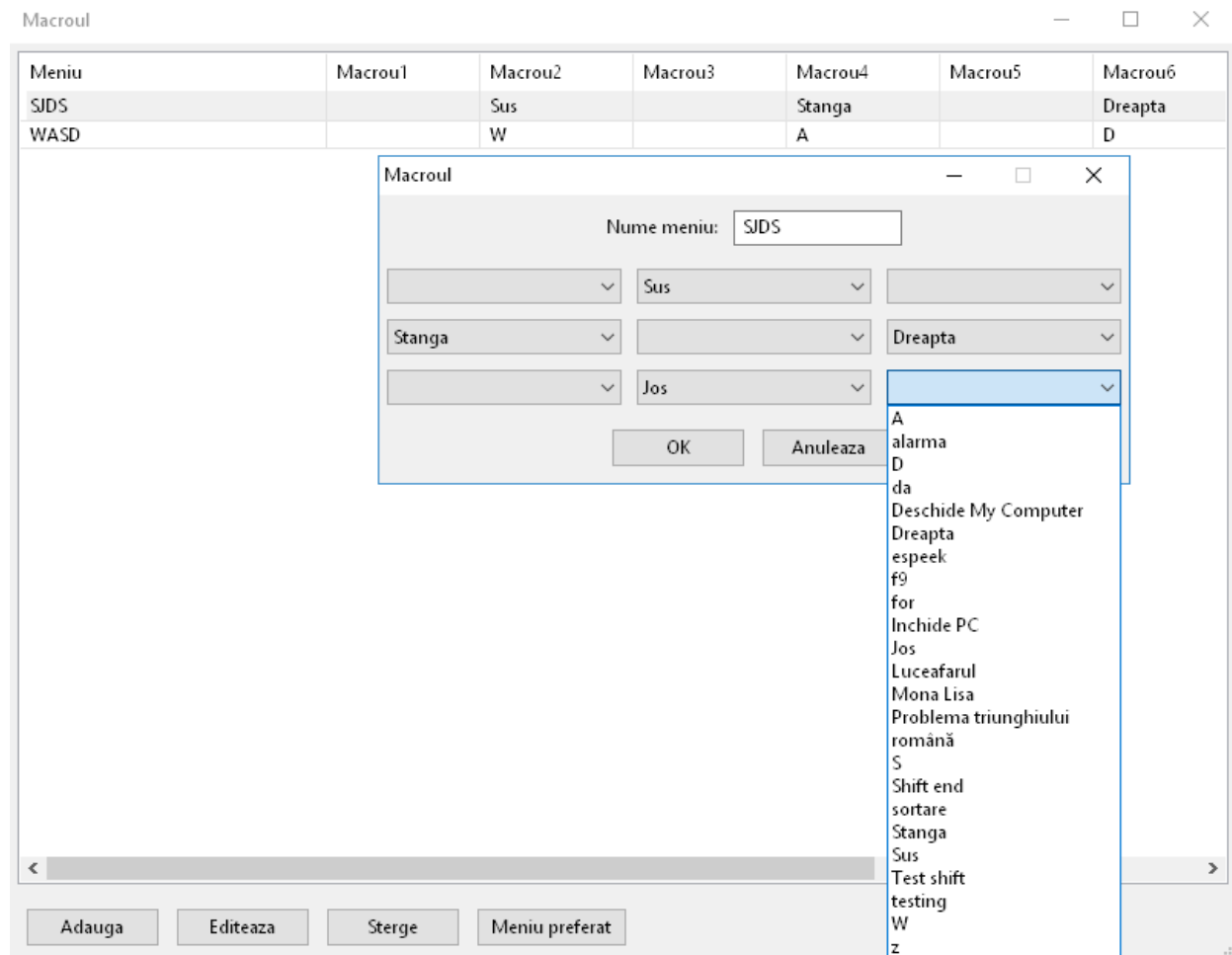
- ☐ Exemplu 1
- ☐ Exemplu 2
- ☒ Laptop
- ☐ Muncă
- ☒ PC

Buttons: OK, Anuleaza

**Figura 2.22.** Dialogul "Grupuri"



- **Meniuri**



**Figura 2.23. Panoul "Meniuri"**

Dacă anumite macroui au legătură între ele (așa cum am văzut în introducere, cele care apasă sus / jos / dreapta / stânga) sau dacă sunt folosite foarte des, este firesc să le grupăm. Un meniu este deci o grupare de maximum 9 macroui (sub forma unui matrix 3x3). Se poate alege și un meniu preferat, acesta o să fie meniul selectat default în website.

### 2.3.19. "Sabloane" (Macroui cu variabile)

O aparentă limitare a macroului în forma prezentată până acum este lipsa inputului. În realitate, există rezolvări dar nu prea ușor de folosit / înțeles. Dăm acum două exemple simple de astfel de rezolvări. Pentru probleme cu număr fix de argumente (suma a două numere) am putea introduce spre exemplu acele variabile într-un fișier .txt din care le putem extrage ulterior astfel:

```
Asteapta(2000)
Taste(O tasta,Ambele,CTRL,HOME)
Asteapta(100)
Taste(O tasta,Ambele,SHIFT,END)
Asteapta(100)
Taste(O tasta,Ambele,CTRL,C)
Asteapta(1000)
Taste(O tasta,Ambele,WIN,)
```

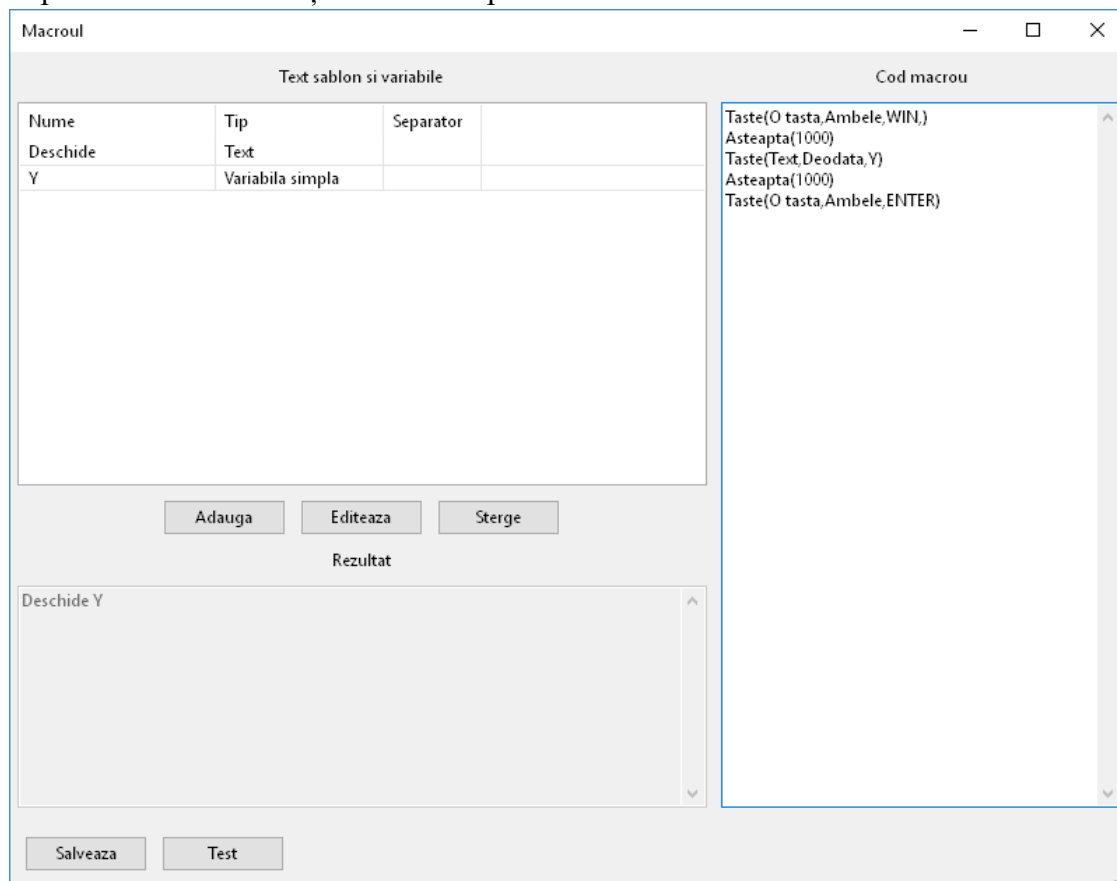
```

Asteapta(1000)
Taste(Text,Interrupt,Calculator)
Asteapta(1000)
Taste(O tasta,Ambele,ENTER)
Asteapta(1000)
Taste(O tasta,Ambele,CTRL,V)
Asteapta(1000)
Taste(O tasta,Ambele,ALT,TAB)
Asteapta(1000)
Taste(O tasta,Ambele,RIGHT)
Asteapta(100)
Taste(O tasta,Ambele,SHIFT,END)
Asteapta(1000)
Taste(O tasta,Ambele,CTRL,C)
Asteapta(1000)
Taste(O tasta,Ambele,ALT,TAB)
Asteapta(1000)
Taste(O tasta,Ambele,+)
Taste(O tasta,Ambele,CTRL,V)
Taste(O tasta,Ambele,ENTER)

```

O altă variantă este executarea unui program care citește un anumit input și eventual pregătește un fișier pentru următorul executabil. Acest lucru presupune ca utilizatorul să cunoască un limbaj de programare sau să se gândească la alte modalități mai ingenioase. Evident aceste metode nu sunt destul de bune pentru a scrie un program repede și clar. A fost deci necesară introducerea unor variabile.

Înainte de a explica panoul pentru Șabloane, amintim că utilizatorul poate, la fel ca la macrouri, să caute și să adauge la lista proprie șabloane făcute de către alți utilizatori care au ales să facă publice macrourile / șabloanele respective.



**Figura 2.24.** Editor șabloane

Editorul de șabloane are 3 părți:

- Text sablon si variabile

Aici se găsește un tabel în care se pot adăuga, șterge și edita Texte, Variabile simple și Variabile multiple. Textul reprezintă partea statică a șablonului. De exemplu, în **Figura 2.24.**, șablonul începe cu textul "Deschide " (cu tot cu spațiu!) și știm că acel text trebuie scris mereu. Variabilele pot fi simple sau multiple. Pentru cele multiple e nevoie și de introducerea unui separator. Un exemplu cu variabile multiple o să urmeze mai jos.

- Rezultat

După fiecare modificare adusă tabelului de mai sus se generează un text ce arată forma actuală a șablonului și modul în care trebuie scris șablonul pentru a fi executat.

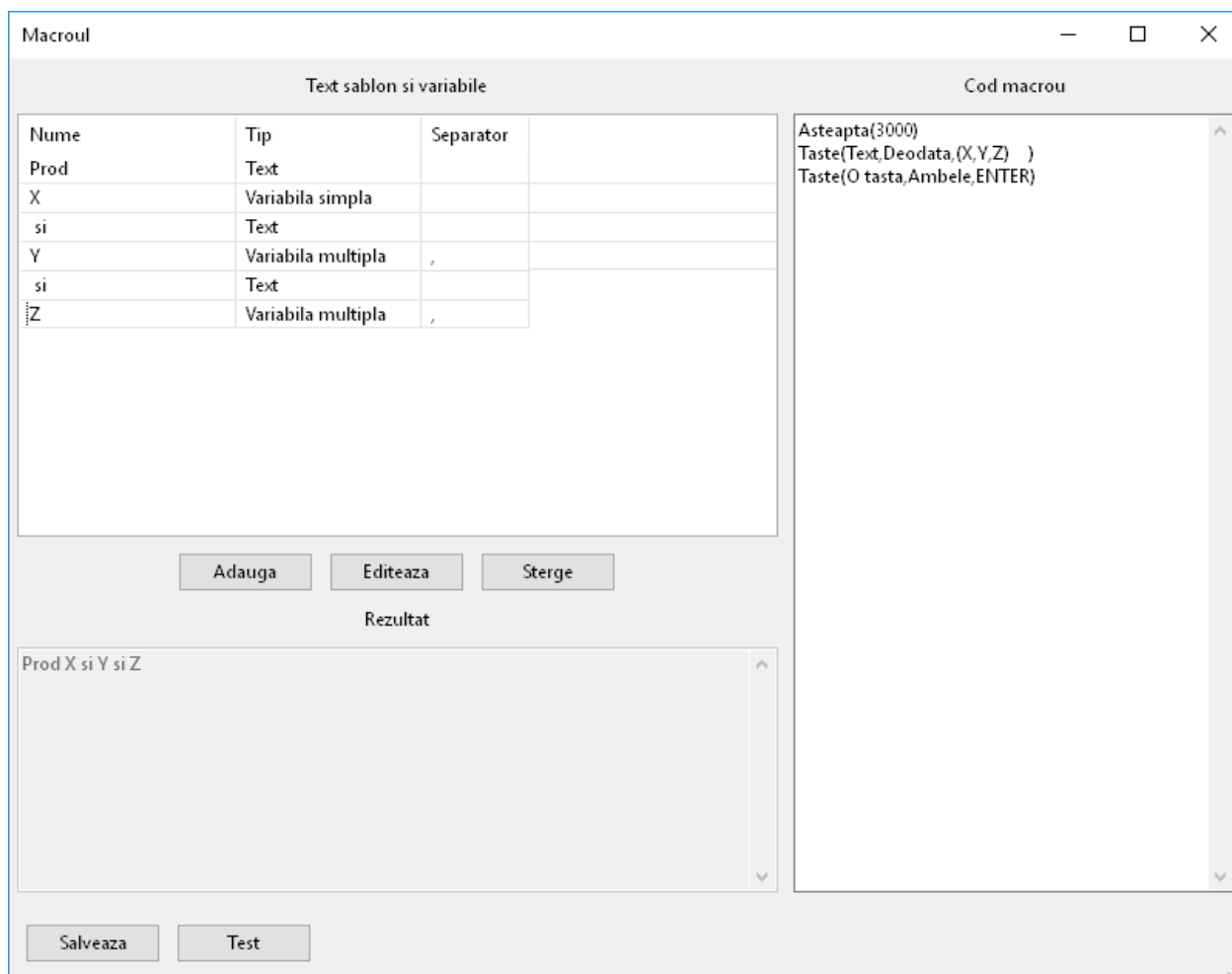
- Cod macrou

În partea dreaptă utilizatorul poate să scrie cod macroul ce urmează a fi executat la apelarea șablonului. Variabilele se pot scrie în partea de cod (de exemplu "Y"-ul de pe linia 3) și o să fie înlocuite la rulare cu inputul dat de utilizator. Pentru variabile multiple, se împarte stringul corespunzător variabilei respective în n părți (folosind separatorul) și apoi se apelează linia din care face parte variabila respectivă folosind pe rând cele n părți. Utilizatorul trebuie să aleagă numele variabilelor astfel încât să nu apară pericolul înlocuirii în alte părți în mod neașteptat (de exemplu dacă în loc de "Y" alegeam ca nume de variabilă "a" programul va avea un comportament nedorit deoarece fiecare "a" o să fie înlocuit și observăm că există unul pe fiecare rând).

Cel care implementează astfel de șabloane trebuie să respecte anumite reguli simple (de care o să fie anunțați dacă nu sunt respectate):

- Să nu fie gol tabelul cu text și variabile;
- Să nu existe două variabile una după alta (trebuie să existe un text pentru a putea separa variabilele între ele, dacă se dorește ca două variabile să fie una după alta, se pot folosi variabilele multiple);
- Șablonul trebuie să înceapă cu un text (pentru a fi unic determinat șablonul respectiv) și acesta trebuie să fie diferit de restul șabloanelor pe care le are utilizatorul (pentru textul T al șablonului curent și pentru orice alt text de început al altui șablon S trebuie respectată regula S nu este prefix al lui T și T nu este prefix al lui S);
- Să nu existe două variabile cu același nume.

Pot exista mai multe variabile pe același rând. În acest caz se generează toate posibilitățile (produs cartezian). Un simplu exemplu este șablonul următor:



**Figura 2.25.** Exemplu de produs cartezian

Putem testa acest șablon cu textul "Prod 0 si 1,2,3 si 4,5,6". După 3 secunde (timp destul pentru a deschide un notepad), aplicația scrie "(0,1,4) (0,1,5) (0,1,6) (0,2,4) (0,2,5) (0,2,6) (0,3,4) (0,3,5) (0,3,6) ".

Uneori nu ne ajunge să folosim o singură instrucțiune pentru o variabilă (de exemplu cum am putea să scriem fiecare tuplu pe câte o linie?). De aceea putem scrie în partea de "Cod macrou" alte șabloane. Dacă dorim să folosim "Matrice [1,2,3;4,5,6;7,8,9]" pentru a scrie într-un fișier .txt matricea respectivă, o posibilă implementare este formată din următoarele șabloane (tabelul se deduce din context) :

- Șablonul Matrice
  - Format: "Matrice [Linie]"
  - Cod macrou:
    - Sablon(Deschide Notepad)
    - Asteapta(1534)
    - Sablon(Scrie linie Linie)
- Șablonul Linie
  - Format: "Scrie linie X"
  - Cod macrou:
    - Sablon(Scrie element X)
    - Taste(O tasta,Ambele,ENTER)

- Șablonul Element

Format: "Scrie element X"

Cod macrou:

Asteapta(5)

Taste(Text,Deodata,X)

Asteapta(5)

Este recomandat ca utilizatorul care dorește să adauge șabloane să facă întâi un macrou asemănător cu ce dorește să introducă în partea de "Cod macrou" a șablonului și să dea export la macroul respectiv. Fișierele se pot deschide cu orice editor de text iar formatul este foarte simplu de înțeles și de urmărit. Fiecare instrucțiune se scrie pe o linie (în cazul textelor, email-urilor, etc. care conțin caracterul '\n', acesta se înlocuiește cu "<n>") astfel: Comentariul liniei respective, urmat de stringul separator "|||" și instrucțiunea. Exemplu:

```
|||Asteapta(1000)
```

```
schimb culoarea|||Asteapta(100)
```

```
|||Taste(Text,Deodata,Exemplu<n>Exemplu <n>)
```

```
|||Taste(Otasta,Ambele,ALT.)
```

```
|||Asteapta(100)
```

```
|||Taste(Otasta,Ambele,H)
```

Dacă este pornită din setări opțiunea "Comanda vocala", se pot apela șabloane direct, folosind cuvântul cheie "șablon" înainte de comanda respectivă. De exemplu "șablon deschide X" o să deschidă X.

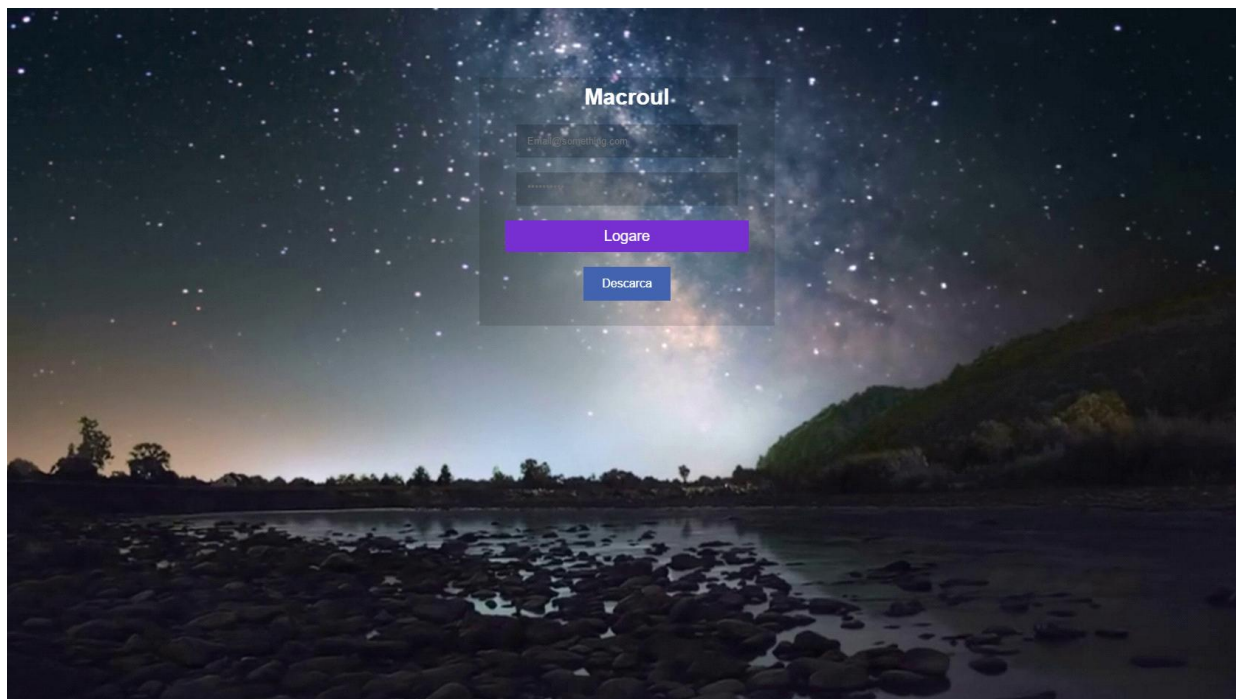
### 2.3.20. "Setari si informatii"

Figura 2.26. "Setari si informatii"

În dialogul "Setari si informatii" vedem ultimele acțiuni pe care le poate face utilizatorul în această aplicație. Aici poate vedea ID-ul lui (pentru ca alții să vadă macrourile / șabloanele acestuia în căutări), poate seta timpul între executarea a două instrucțiuni macro succesive, portul COM pentru cititorul de carduri, dacă vrea să introducă mereu parola sau nu. Ultimele câmpuri se referă la thread-urile care rulează, de exemplu dacă nu se folosesc mouse-ul și tastatura ca activatori, nu are sens să se urmărească activitatea acestora. Partea de "Stream imagine" o să capete sens în prezentarea website-ului.

## Capitolul 3. Website

În anul 2010 smartphone-urile aveau mai multe vânzări în lume decât PC-urile. În 2016 mai multe website-uri erau accesate de pe telefoane / tablete decât de pe PC-uri. A devenit astfel evident faptul că trebuiau introduse și pentru telefoane anumite facilități. Telefonul devine astfel pentru această aplicație o telecomandă pentru PC-uri.



**Figura 3.1. [www.macroul.xyz](http://www.macroul.xyz)**

Intrând pe website-ul aplicației ([www.macroul.xyz](http://www.macroul.xyz)) utilizatorul poate descărca aplicația (este un .zip ce trebuie dezarhivat, nu e nevoie să fie nimic instalat) sau să se logheze cu aceleași date de logare ale aplicației (email și parolă).

După autentificare, utilizatorul vede în partea de sus "meniul preferat" (se alege din aplicație meniul care să apară default) sau poate să aleagă alt meniu care să apară în locul acestuia. La apăsarea unui buton din meniu se trimite comanda (macroul) direct către toate calculatoarele care au fost selectate să se activeze.

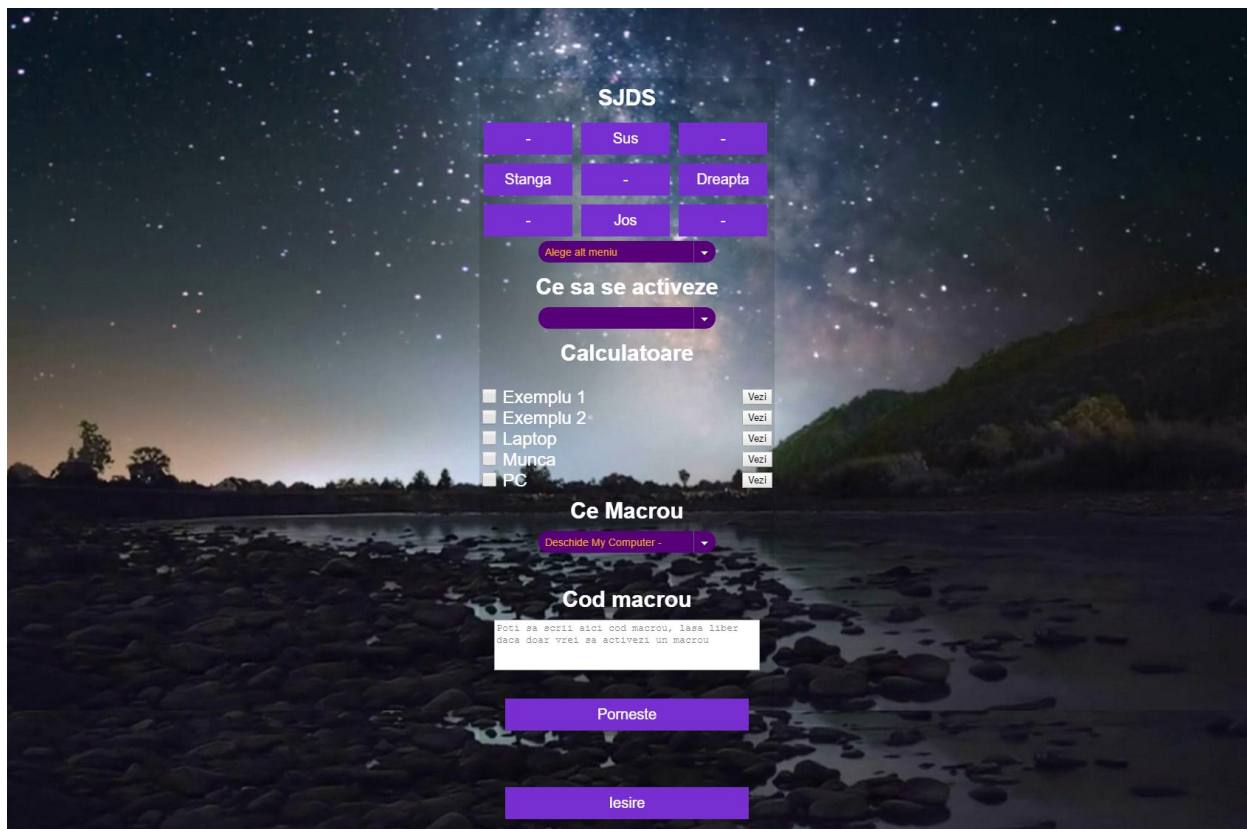
Pentru activare există 3 opțiuni: Toate (toate calculatoarele utilizatorului care au un nume definit), Grupuri și Calculatoare (pentru ultimele două se aleg care elemente din acea listă să fie activate cu ajutorul checkbox-urilor).

Apoi se alege unul dintre macroulurile definite de utilizator sau se poate scrie unul ad-hoc prin scrierea instrucțiunilor în textbox-ul "Cod macroul". După ce e totul pregătit, se apasă pe butonul "Pornește" și macroul ales (xor - sau exclusiv - instrucțiunile tocmai scrie) o să fie executat.

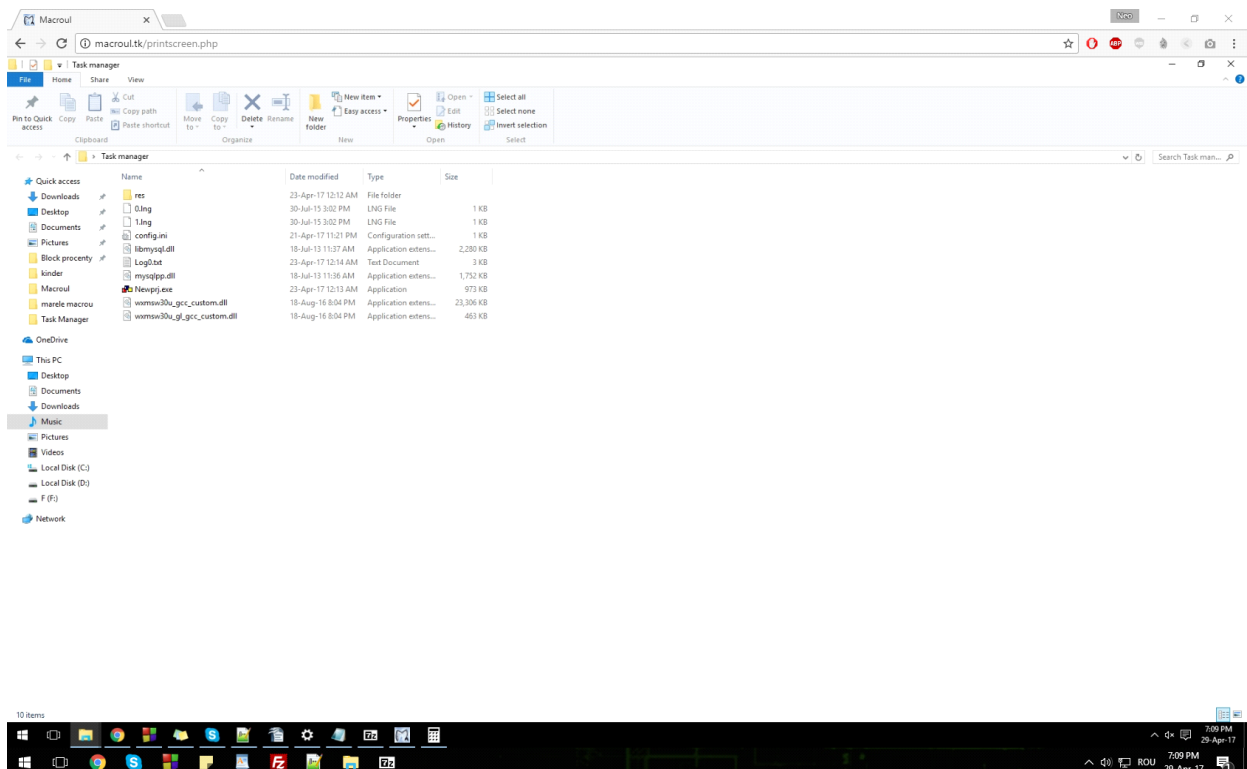
Atenție, pentru meniuri nu se apasă pe butonul "Pornește".

Observăm că atunci când se alege opțiunea "Calculatoare" (când alegem ce să se activeze) apare în dreptul fiecărui calculator butonul "Vezi". Acesta ne oferă posibilitatea să

vedem calculatorul respectiv și să folosim direct din browser calculatorul respectiv așa cum folosim calculatorul nostru (se trimit inputurile de tastatură și mouse).



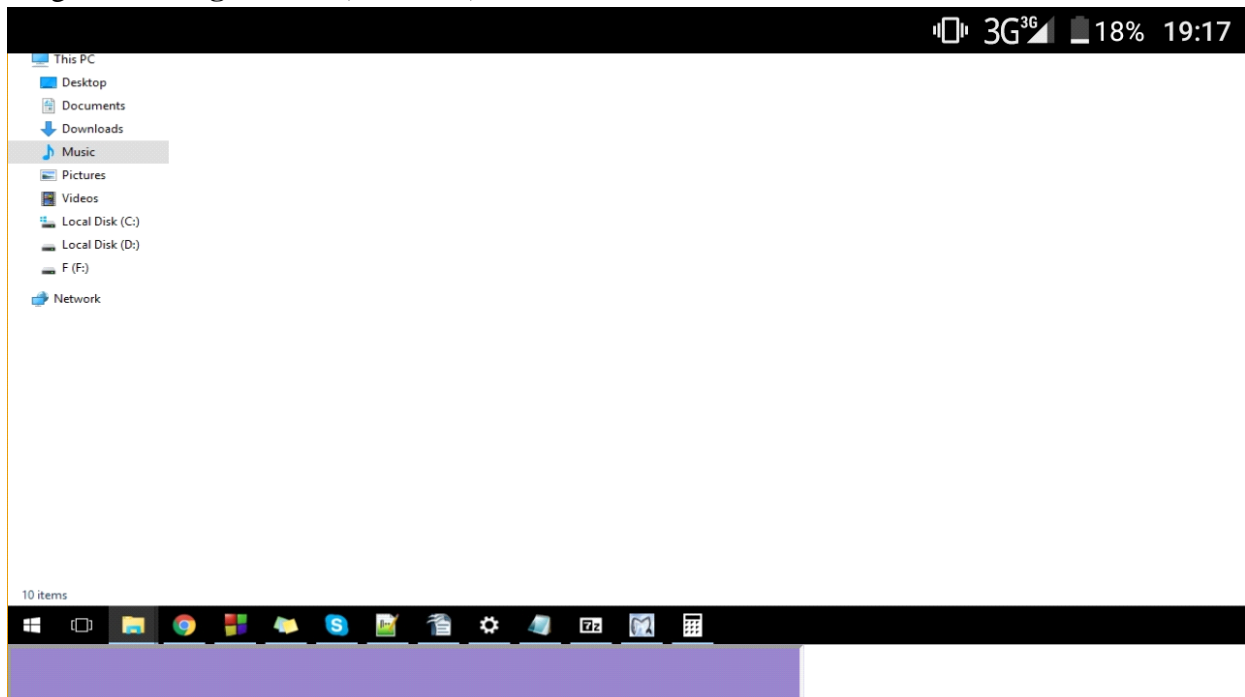
**Figura 3.2. Meniu principal**



**Figura 3.3. Opțiunea "Vezi " de pe un calculator**



Pentru telefoane este la fel de simplu, singura diferență fiind modul în care se introduce textul. Pentru a folosi tastatura se introduce textul dorit într-un input special ce se află sub imagine ca în **Figura 3.4.** (bara mov).



**Figura 3.4.** Opțiunea "Vezi " de pe un telefon

### Detalii implementare:

Aplicația crează o poză cu ecranul (! dacă acest lucru se dorește !) la un interval de timp ales de utilizator în setări, o redimensionează și schimbă calitatea pozei în funcție de setările alese, salvează această imagine și suprascrive orice imagine era în baza de date pentru acel calculator (deci la orice moment există maximum o astfel de poză salvată).

Pe partea de PHP, se actualizează sursa imaginii afișate făcând o cerere POST la un alt fișier care se conectează la baza de date și furnizează cu ce ar trebui înlocuită sursa:

```
function poza()
{
    $.post("ImgAjax.php",function(data){
        $("#id").prop("src",data);
        setTimeout(poza,333);
    });
}
poza();
```

Iar în ImgAjax.php:

```
$sql = "SELECT `Poza` FROM ... ";
$result = mysqli_query($con,$sql);
if($result && $result->num_rows>0)
    $res=$result->fetch_assoc()['Poza'];
echo 'data:image/jpeg;base64,'.base64_encode( $res);
```

Pentru comenzile de click de exemplu am adăugat un "EventListener" pentru body care verifică dacă este un click sau un dublu click (așteaptă încă 222 de milisecunde în care vede dacă s-a mai făcut un click sau nu) și apoi inserează în baza de date informația că s-a făcut click, dublu click sau click dreapta. Pentru tastatură există la fel două funcții handle (pentru keydown și keyup) care introduc în baza de date evenimentele în ordinea în care au apărut, ceea ce permite trimiterea unor combinații de taste. După ce sunt executate în aplicație, aceste informații se șterg.

## Capitolul 4. Exemple

- **Testare și interfețe**

- La locul de muncă, exista un bug care, la apăsarea unui buton (după un număr aleator de ori) închidea aplicația. După rezolvarea acelui bug, aplicația trebuia testată, astfel a fost firesc să folosesc macroul să facă acele 9 click-uri de cât mai multe ori. După 4 ore de testare automată (600 de rulări a macroului următor), aplicația încă mergea.

```
Asteapta(3000)
Click(558,702,Stanga,Ambele)
Asteapta(3000)
Click(790,413,Stanga,Ambele)
Asteapta(2000)
Click(537,423,Stanga,Ambele)
Asteapta(2000)
Click(535,853,Stanga,Ambele)
Asteapta(2000)
Click(951,707,Stanga,Ambele)
Asteapta(2000)
Click(585,688,Stanga,Ambele)
Asteapta(2000)
Click(589,315,Stanga,Ambele)
Asteapta(2000)
Click(846,720,Stanga,Ambele)
Asteapta(4000)
Click(948,711,Stanga,Ambele)
Asteapta(2000)
```

- Tot la locul de muncă, după deschiderea aplicației, pentru a ajunge în panoul în care lucrez trebuie să mă autentific, eventual trebuie ajuns în alt panou pentru a reseta starea curentă a aplicației și apoi o altă succesiune de butoane ce trebuie apăstate pentru a ajunge la panoul dorit. Cum dimensiunile ferestrelor nu se schimbă (sau dacă se întâmplă acest lucru, ele pot fi redimensionate și repositionate), este preferată alegerea apăsării unei singure taste care execută mai repede și fără greșală acea listă de comenzi.

- **Jocuri**

- Pentru jocuri, se disting două posibilități:
  - Pentru cele "supravegheate", se pot crea macrouri care fiind activate, execută comenzi de taste și mouse foarte rapide, apoi jucătorul continuă singur.
  - Pentru cele "nesupravegheate", folosind eventual și comanda "Asteapta\_schimbare" se pot crea macrouri mai complexe, având mult mai multe rânduri care sunt făcute special să ruleze mai multe ore. Acestea, fiind nesupravegheate, trebuie bine gândite astfel încât la început să fie jocul într-o stare cunoscută și mai ales la sfârșit să revină la o stare de unde poate să o ia de la capăt.

- **Programare**

- Macroul "for"

```
Taste(O tasta,Ambele,CTRL,LEFT)
Taste(Text,Deodata,for(int a=0;a<)
Taste(O tasta,Ambele,CTRL,RIGHT)
Taste(Text,Deodata,.size();a++))
Taste(O tasta,Ambele,ENTER)
Taste(Text,Deodata,{ })
Taste(O tasta,Ambele,LEFT)
```

```
Taste(O tasta,Ambele,ENTER)
Taste(O tasta,Ambele,CTRL,LEFT)
Taste(O tasta,Ambele,ENTER)
```

Care transforma nume\_variabila în

```
for(int a=0;a<nume_variabila.size();a++)
{
}
}
```

-Macroul "sortare"

```
Taste(O tasta,Ambele,CTRL,SHIFT,LEFT)
Taste(O tasta,Ambele,CTRL,C)
Taste(O tasta,Ambele,RIGHT)
Taste(O tasta,Ambele,SPACEBAR)
Taste(O tasta,Ambele,CTRL,V)
Taste(O tasta,Ambele,SPACEBAR)
Taste(O tasta,Ambele,CTRL,V)
Taste(O tasta,Ambele,SPACEBAR)
Taste(O tasta,Ambele,CTRL,V)
Taste(O tasta,Ambele,SPACEBAR)
Taste(O tasta,Ambele,CTRL,V)
Taste(O tasta,Ambele,SPACEBAR)
Taste(O tasta,Ambele,CTRL,V)
Taste(O tasta,Ambele,SPACEBAR)
Taste(O tasta,Ambele,CTRL,LEFT)
Taste(O tasta,Ambele,CTRL,LEFT)
Taste(O tasta,Ambele,CTRL,LEFT)
Taste(O tasta,Ambele,CTRL,LEFT)
Taste(O tasta,Ambele,CTRL,LEFT)
Taste(O tasta,Ambele,CTRL,LEFT)
Taste(Text,Deodata,for(int a=0;a<)
Taste(O tasta,Ambele,CTRL,RIGHT)
Taste(O tasta,Ambele,BACKSPACE)
Taste(Text,Deodata,.size();a++))
Taste(O tasta,Ambele,ENTER)
Taste(Text,Deodata,for(int b=a+1;b<)
Taste(O tasta,Ambele,CTRL,RIGHT)
Taste(O tasta,Ambele,BACKSPACE)
Taste(Text,Deodata,.size();b++))
Taste(O tasta,Ambele,ENTER)
Taste(Text,Deodata,if()
Taste(O tasta,Ambele,CTRL,RIGHT)
Taste(O tasta,Ambele,BACKSPACE)
Taste(Text,Deodata,[a]<)
Taste(O tasta,Ambele,CTRL,RIGHT)
Taste(O tasta,Ambele,BACKSPACE)
Taste(Text,Deodata,[b]))
Taste(O tasta,Ambele,ENTER)
Taste(Text,Deodata,swap()
Taste(O tasta,Ambele,CTRL,RIGHT)
Taste(O tasta,Ambele,BACKSPACE)
Taste(Text,Deodata,[a],)
Taste(O tasta,Ambele,CTRL,RIGHT)
Taste(Text,Deodata,[b]);)
Taste(O tasta,Ambele,ENTER)
```

Care transforma nume\_variabila în

```
for(int a=0;a<nume_variabila.size();a++)
    for(int b=a+1;b<nume_variabila.size();b++)
        if(nume_variabila[a]<nume_variabila[b])
            swap(nume_variabila[a],nume_variabila[b]);
```

- **Prezentari/predare**

- Macroul este o unealtă foarte utilă pentru prezentări, pentru persoane foarte bine organizate se poate face de exemplu un macrou simplu care schimbă slide-urile prezentării după

un număr fix de secunde. De asemenea, având acel meniu care se poate accesa din browser, este foarte bun pentru a face o demonstrație live care a fost pregătită și astfel nu vor apărea surprize.

- Un exemplu de macro care prezintă problema triumfului de la programare dinamică este următorul (deschide un website unde este descrisă problema, apoi după un număr oarecare de secunde, în cazul acesta 20, se deschide un mediu de dezvoltare, în cazul acesta Code::Blocks, se crează un proiect nou, se introduce codul sursă care rezolvă problema, se crează un fișier cu date și se execută programul):

```

|||Website(http://olidej.wikispaces.com/file/view/Suma%20in%20triumphi.pdf/472125932/Suma%20in%20triumphi.pdf)
Asteapta 20s|||Asteapta(20000)
|||Sablon(Deschide Code)
|||Asteapta(40000)
|||Taste(Otasta,Apasa,ALT,)
|||Asteapta(100)
|||Taste(Otasta,Ambele,F)
|||Asteapta(100)
|||Taste(Otasta,Ridica,ALT,)
|||Asteapta(100)
|||Taste(Otasta,Ambele,ENTER)
|||Asteapta(500)
|||Taste(Otasta,Ambele,P)
|||Asteapta(3000)
click console app, posibil sa nu mearga pe alte rezolutii|||Click(707,235,Stanga,Ambele)
|||Asteapta(1000)
|||Taste(Otasta,Ambele,ENTER)
|||Asteapta(1000)
|||Taste(Otasta,Ambele,ENTER)
|||Asteapta(3000)
|||Taste(Text,Deodata,PD)
|||Asteapta(1000)
|||Taste(Otasta,Ambele,ENTER)
|||Asteapta(1000)
|||Taste(Otasta,Ambele,ENTER)
|||Asteapta(5000)
click sources|||Click(34,158,Stanga,Ambele)
|||Asteapta(1000)
|||Click(71,176,Stanga,Ambele)
|||Asteapta(50)
|||Click(71,176,Stanga,Ambele)
|||Asteapta(4000)
|||Asteapta(1000)
click undeva in cpp|||Click(1178,310,Stanga,Ambele)
|||Taste(Otasta,Ambele,CTRL,A)
|||Taste(Text,Deodata,#include<fstream><nl>#include<iostream><nl>using namespace std;<nl>int n,m,T[101][101],S[101][101]; ...)
|||Taste(Otasta,Ambele,CTRL,SHIFT,N)
|||Asteapta(1000)
|||Taste(Otasta,Ambele,ENTER)
|||Taste(Text,Deodata,triumphi.in)
|||Taste(Otasta,Ambele,ENTER)
|||Taste(Otasta,Ambele,ENTER)
|||Asteapta(1000)
|||Taste(Text,Deodata,5<nl>7<nl>3 8<nl>8 1 0<nl>2 7 4 4<nl>4 5 2 6 5)
|||Asteapta(100)
|||Taste(Otasta,Ambele,F9)

```

- **Acces de la distanță**

-Cum am mai spus, Macroul poate să fie folosit și ca o telecomandă, având acele meniuri la care se adaugă posibilitatea executării unui macro nou definit dar și partea de acces la distanță. Au fost destule situații în care a fost nevoie să văd ce informații conțin anumite fișiere la locul de muncă, iar faptul că puteam să le răspund în câteva minute folosind doar telefonul

(fără acces la un calculator) a fost foarte util. De asemenea, pentru a schimba o piesă muzicală / video sau chiar pentru a te juca (având cele 4 macroui (sus, jos, dreapta, stânga) este suficient pentru jocuri simple precum pac-man, tetris etc.) este mai mult decât suficient.

- **Programe "command line"**

-Se pot adăuga oricâte programe care să fie chemate în macrou folosind instrucțiunea Executa. Acestea se transformă cu ușurință în șabloane pentru a simplifica scrierea. Un exemplu sugestiv îl consider a fi acela în care se dorește rostirea unui text dat. Un exemplu de macrou este:

```
Executa(C:\Program Files (x86)\eSpeak\command_line\espeak.exe,Async,Arata,-vro+f3 "Exemplu")
```

Dacă această instrucțiune este folosită de mai multe ori, se poate scrie următorul șablon care are forma "Vorbeste X" (cu X variabilă) ce are codul macrou asociat:

```
Executa(C:\Program Files (x86)\eSpeak\command_line\espeak.exe,Async,Arata,-vro+f3 "X")
```

După care se poate apela șablonul: Sablon(Vorbeste Exemplu) și ar avea același efect ca macroul de mai sus.

- **Automatizări pentru probleme specifice**

-Evident, principalul scop al aplicației este acela de a automatiza cât mai multe rutine, deci este esențial să fie cât mai generic. Sper că, ajungând până aici, cititorul să găsească o utilizare a macroului și să poată rezolva singur acea problemă folosind ca punct de plecare exemplele de mai sus.

## Capitolul 5. Provocări

- **Tastele în android.**

Am vrut neapărat să se poată folosi tastatura telefonului pentru a trimite taste în timp real calculatorului (așa cum era deja în browser-ul de pe desktop). După ce am reușit să deschid tastatura telefonului, am observat că spre deosebire de varianta de desktop (în care fiecare literă avea un cod, de exemplu 16 pentru shift și 65 pentru 'a'), pe telefon toate tastele aveau codul 229. Explicația dată de ei este că acea literă care a fost scrisă poate să fie ulterior înlocuită când se folosește autocorrect-ul. Soluția a fost să creez acel spațiu mov deja menționat și ca la apăsarea tastei Enter să se trimită ce a fost scris până atunci (tasta Enter are mereu codul 13).

- **Folosirea diacriticelor**

Aplicația nu a fost scrisă cu diacritice însă am crezut necesar să se poată scrie astfel în macrou (de exemplu în comanda Taste). Connector-ul de MySQL pe care l-am folosit pentru comunicarea cu baza de date nu are idee de unicode deci a trebuit să găesc o altă soluție ("MySQL++ currently doesn't have any code in it for Unicode conversions; it just passes data along unchanged from the underlying MySQL C API, so you still need to be aware of these underlying issues."). Pentru rezolvarea acestei probleme a fost nevoie să folosesc două funcții windows (WideCharToMultiByte și MultiByteToWideChar) care simulează wchar\_t-urile.

Exemplu de macrou cu diacritice:

```
Sablon(Deschide Notepad)
Asteapta(2000)
Taste(Text,Deodata,Mihai Eminescu<nl>Luceafărul <nl><nl>A fost odată ca-n povești,<nl>A fost ca niciodată.<nl>Din rude mari
împărătești,<nl>O prea frumoasă fată)
```

- **Acces la distanță**

Probabil această parte a durat cel mai mult să fie implementată. Prima problemă a fost generarea de printscreen-uri. Aveam deja o funcție care făcea printscreen și salva acea imagine, dar se salva ca BMP. De exemplu, o astfel de imagine are pe rezoluția mea curentă (1680 x 1050) 5 MB, ceea ce era mult prea mult pentru a trimite câteva zeci de astfel de imagini pe minut, de exemplu către telefoane (unde contează foarte mult câtă baterie folosești, cât internet etc.). Apoi o soluție a fost să transform acea imagine în una JPEG folosind un tool de tip command line, dar tot trebuia să scriu acea poză 5 MB pe disk, apoi să o transform în JPEG și tot așa. Până la urmă am reușit să transform captura de imagine direct în JPEG, iar utilizatorul poate alege calitatea imaginii (pentru 10% calitate, se ajunge la imagini de 75 KB care sunt mult mai ușor de prelucrat ulterior). Pe partea de website, problema era faptul că imaginea era pusă în cache și nu se reactualiza când se schimba fișierul la care se uita. Când am forțat reactualizarea imaginii aceasta se schimba, însă fiecare imagine care se afișa era pusă în cache iar după câteva zeci de afișări pagina nu mai funcționa. Soluția în acest caz a fost să citesc din baza de date bloburi, să le codific (în baza 64 folosind funcția php "base64\_encode") și să le afișez folosind apeluri Ajax cu pauză de 0.333 secunde. Pentru transferul între calculatorul în care este deschis macroul și server am încercat să folosesc FTP pentru un transfer mai rapid, dar s-a dovedit că nu era mereu cazul, au fost situații în care era nevoie de peste 10 secunde pentru conectare și transfer (spre deosebire de a trimite ca blob în SQL unde era de ajuns o secundă). O altă

problemă folosind FTP-ul a fost faptul că existau situații în care se scria și se citea din același fișier concomitent, astfel imaginile ajungeau să fie modificate și neclare. Am mai încercat o combinație între FTP și MySQL în care trimiteam imaginea prin FTP atunci când era anunțat în MySQL că se dorește acest lucru. În realitate, nu se observa nici o diferență deoarece atunci când inseram în baza de date faptul că am trimis fișierul și acesta poate să fie folosit, dura la fel de mult ca atunci când trimiteam direct fișierul. De ce? Pentru că majoritatea timpului în ambele cazuri era folosit pentru a se face conexiunea la MySQL.



## Capitolul 6. Direcții de îmbunătățire

Programul este securizat și optimizat la un nivel mediu, suficient din punctul meu de vedere. Un exemplu de optimizare ce se poate aduce ar putea să fie modul în care se trimit imaginile de la calculator la server. În loc să se trimită mereu poze complete, o abordare asemănătoare videourilor ar putea să fie mai utilă (să se trimită cât mai comprimat doar diferențele față de ultima imagine). Partea de instrucțiuni o consider suficientă deoarece, așa cum am arătat în exemple, se pot adăuga oricâte alte funcționalități, însă pentru activatoare ar putea exista mai multe posibilități (senzori de lumină, sunet, temperatură etc.). De cele mai multe ori, utilizatorul nu dispune de aparatură mai sofisticată sau suficient de răspândită pentru a implementa noi activatori. Cum aplicația este open-source, fiecare poate să își adauge propriul thread care ascultă la un eventual nou dispozitiv.

## Capitolul 7. Concluzii

Amintim în încheiere principalele avantaje ale macroului față de alte aplicații asemănătoare:

- Este unul dintre puținele programe de macrouri în limba română (dacă nu chiar singurul);
- Este unul dintre puținele programe (dacă nu chiar singurul) care salvează macrourele într-o bază de date pentru ca utilizatorul să aibă acces de oriunde la ele, fără a avea zeci sau chiar sute de fișiere care trebuie păstrate;
- Existența unei interfețe ușor de folosit. În multe aplicații asemănătoare, utilizatorul ori trebuie să scrie cod într-un editor text așa cum este cazul AHK (în acest caz este dificil pentru cei care nu sunt programatori) ori îi sunt înregistrate mișcările mouse-ului și tastaturii și timpul dintre acțiuni (caz în care utilizatorul este limitat la aceste comenzi);
- Este simplu de folosit (14 instrucțiuni simple și intuitive față de cele peste 300 pe care le are AHK), fiecare instrucțiune fiind clară, având câte un exemplu și o argumentare pentru existența acelei instrucțiuni;
- Este unul dintre puținele programe (dacă nu chiar singurul) care permite activarea macrourilor (deja definite sau chiar unele noi) de la distanță;
- O schimbare a unui macrou folosind aplicația, duce la modificarea tuturor calculatoarelor care folosesc aplicația sub același utilizator. Aceste schimbări pot apărea zilnic și este important să nu se piardă timpul verificând manual fiecare calculator în ce stadiu este și apoi aducerea lui la zi;
- Posibilitatea de a adăuga un macrou făcut de alții direct din aplicație;
- Posibilitatea de a genera cod macrou folosind alte limbaje de programare (de exemplu pentru a testa asta, am creat în C++ un fișier .Macrou care are peste 2000 de linii și reușește să deseneze Mona Lisa în paint folosind 3 culori);
- Nu trebuie instalat și nici nu depinde de alte programe;
- Este gratuit;
- Este open-source.

## Capitolul 8. Link-uri utile

- [1] <https://pypi.python.org/pypi/google#downloads>
- [2] <http://pythonhosted.org/google/>
- [3] <https://pypi.python.org/pypi/beautifulsoup4>
- [4] <http://www.pajamaprogrammer.com/2015/11/convert-python-35-script-py-to-windows.html>
- [5] [https://github.com/Uberi/speech\\_recognition/blob/master/reference/library-reference.rst](https://github.com/Uberi/speech_recognition/blob/master/reference/library-reference.rst)
- [6] <https://pypi.python.org/pypi/SpeechRecognition/>
- [7] <https://github.com/muquit/mailex/releases>
- [8] <https://perrohunter.com/how-to-kill-a-process-by-its-pid-on-windows/>
- [9] <http://stackoverflow.com/questions/7956519/how-to-kill-processes-by-name-win32-api>
- [10] <http://stackoverflow.com/questions/1264137/how-to-copy-string-to-clipboard-in-c>
- [11] <http://stackoverflow.com/questions/17638674/how-to-wait-for-shellexecute-to-run>
- [12] <http://stackoverflow.com/questions/27176241/beginthreadex-with-member-functions>
- [13] <http://www.dreamincode.net/forums/topic/92806-uploading-to-an-ftp-server/>
- [14] <https://www.youtube.com/watch?v=sc3azoiDilM>
- [15] <http://planetsourcecode.com/vb/scripts/ShowCode.asp?txtCodeId=10754&lngWId=3>
- [16] <http://blog.microtom.net/os-management/convert-bmp-to-jpg-command-line-utility>
- [17] <http://stackoverflow.com/questions/1077041/refresh-image-with-a-new-one-at-the-same-url>
- [18] <http://stackoverflow.com/questions/5497073/how-to-differentiate-single-click-event-and-double-click-event>
- [19] <http://stackoverflow.com/questions/13179410/check-whether-one-specific-process-is-running-on-windows-with-c>
- [20] <http://stackoverflow.com/questions/19215637/navigate-back-with-php-form-submission>
- [21] <http://www.cplusplus.com/forum/windows/26987/>