 Universitatea Politehnica Timişoara

Facultatea de Automaticǎ şi Calculatoare

Departamentul de Automatică şi Informatică Aplicată

**Aplicație Desktop Și Mobliă Pentru Fotografierea, Editarea și Afișarea Cursurilor Proiectate**

**PROIECT DE DIPLOMĂ**

Coordonator știinţific: Autor:

Ş. l. Dr. Ing. Ovidiu BANIAŞ Paul-Florin TARCE

Timișoara,

Iunie 2018

CUPRINS

[Capitolul 1.Introducere 3](#_Toc517617473)

[1.1 Motivatie și context 3](#_Toc517617474)

[1.2 Obiective, descrierea, domeniul și utilitatea temei 4](#_Toc517617475)

[1.3 Structura Lucrării 6](#_Toc517617476)

[Capitolul 2.Tehnologii folosite 7](#_Toc517617477)

[2.1 Platforma Microsoft .NET Framework 7](#_Toc517617478)

[2.2 Programarea orientată pe eveniment 12](#_Toc517617479)

[2.3 Biblioteci speciale: AForge, OpenCV 14](#_Toc517617480)

[2.4 Microcontrolerul Arduino Uno. Modulul Bluetooth 16](#_Toc517617481)

[2.5 Android, Java și Android Studio 21](#_Toc517617482)

[2.6 Testarea 24](#_Toc517617483)

[2.7 Protocolul SMTP 25](#_Toc517617484)

[Capitolul 3.Specificațiile aplicației 26](#_Toc517617485)

[3.1 Schema-bloc a sistemului. 26](#_Toc517617486)

[3.2 Funcțiile sistemului 28](#_Toc517617487)

[3.3 Interfața cu utilizatorul 29](#_Toc517617488)

[3.4 Structuri de baze de date și fișiere 34](#_Toc517617489)

[3.5 Baza materială necesară realizării și testării sistemului 35](#_Toc517617490)

[Capitolul 4. Proiectarea Aplicației 37](#_Toc517617491)

[4.1 Arhitectura. Descrierea componentelor și a comunicării între ele 37](#_Toc517617492)

[4.2. Cablaje, desene de echipare, alocarea la pinii conectorilor 41](#_Toc517617493)

[Capitolul 5. Realizarea și Utilizarea Sistemului 43](#_Toc517617494)

[Capitolul 6. Concluzii 50](#_Toc517617495)

[Capitolul 7. Bibliografie 52](#_Toc517617496)

# Capitolul 1.Introducere

## Motivatie și context

Automatizarea unor procese și realizarea de software care să vină în ajutorul omului în diferite domenii și situații, reprezintă unul din cei mai importanți factori, piloni, de altfel ai evoluției tehnologice și științifice din ultimii ani. Cum rolul acestor programe este de a face viata omului mai ușoară și munca acestuia mai eficientă, acest proiect reprezintă un început pentru un nou mod de lucru al posibililor utilizatori.

Proiectul abordat în această lucrare vine în ajutorul cadrelor didactice și studenților din cadrul facultății de Arhitectură și Urbanism din Timișoara, dar aplicația poate să fie folosită în viitor în domeniul educațional și la universități de alt profil sau la alte școli. Proiectul acesta este un început a ceea ce se dorește a fi în viitor un suport pentru un nou mod de predare, care să eficientizeze mai mai mult munca profesorilor și care implicit va avea efect asupra randamentului elevilor în timpul orelor.

Contextul acutal arată că pe măsură ce tot mai multe aspecte din arhitectură, design și fabricație sunt transferate în mediul digital, modul în care acestea sunt predate devine un subiect de importanță mare. Tranziția de la vechile modalități de predare la a celor noi este un subiect discutabil. Una din cauzele principale pentru care acest proiect a început a fost trecerea de la tablele negre(cu cretă) la cele albe(cu markere). În ultimii 15 ani, în Facultatea de Arhitectură și Urbanism din „Universitatea Politehnica Timișoara” modul în care se predă s-a schimbat mult datorită evoluției tehnologice [1]. Schimbarea tablelor a avut unele efecte negative în cadrul cusurilor. Câteva din problemle apărute în urma acestei schimbări vor fi prezentate în continuare, pentru a aduce o justificare clară a necesității unei aplicații care să vină în ajutorul celor afectați de această schimbare. De asemena, aplicația aduce și funcționalități suplimentare neprevăzute neaparat în cerințe, dar benefice.

Din comportamentul studenților la aceste cursuri de design s-a observat că tot mai puțini din ei iau notițe, ei bazându-se pe discursul profesorului și pe pozele cu notițele de pe tablă. Aceste poze sunt distribuite între ei pe rețelele de socializare. Pe de altă parte s-a observat la profesori obiceiul de a îngrămădi multe informații referitoare la o problemă pe o singură tablă, ceea ce arată nevoia de a fotografia tabla și de a salva pozele. O altă problemă este faptul că o tablă albă pe un perete alb nu mai reprezintă centrul de interes la fel cum îl reprezenta o tablă neagră.[1] De asemenea tablele albe au o suprafață care din anumite unghiuri reflectă lumina, cele negre reflectând doar când sunt ude. Grosimea și densitatea liniei cretei pot fi modificate mult mai usor decât cea a markerului. Foarte important penrtu aceste cursuri este lucrul cu mai multe variante de desen. Pe tabla neagră acest lucru era mai usor de făcut pentru că dacă un desen este șters ușor cu buretele(uscat), pe tablă rămân urme superficial ale desenului astfel putându-se face mai multe versiuni sau extinde.

Deși există încercări de a muta tablele în zona digitală interactivă, acestea s-au dovedit a lua mai mult timp decât echivalentul lor analogic. De aceea se dorește o metodă mai convenabilă și eficientă de lucru, posibilă doar prin mijloace hibride(analogic – digital). Schițatul figurilor poate să rămână și este mai de folos să fie făcut manual în timp partea digitală(aplicația) completează procesul de predare[1].

Luând în considerare cele de mai sus , se doreste o digitalizare parțială a procesului de predare. Modul în care se va realiza acest lucru este cu o aplicație desktop(PC) legată cu o aplicație mobilă prin care se vor face poze la tabla, edita și apoi afișa.

## Obiective, descrierea, domeniul și utilitatea temei

Sistemul care a fost construit pentru situațiile prezentate în capitolul anterior face parte din mai multe domenii din IT. El este alcatuit din 3 părți sau subsisteme care sunt: aplicația desktop(PC) cu camera foto, aplicația mobilă și aplicația pe microcontroler. Domeniul în care se încadrează aplicația este preponderent cel al aplicațiilor desktop care se foloseste de anumite periferice, precum camera foto, microcontrolerul Arduino, pentru a avea o funcționalitate specifică și pentru a fi de folos și usor de folosit.

Tema constă în realizarea fotografiilor cu ajutorul unei camere foto sau video conectate la PC, editarea acestora și afișarea lor prin videoproiector pe tabla pe care se predă un anumit curs. Aplicația desktop va putea fi utilizată în mare parte, pentru ușurință, folosind câteva taste prin care se pot face urmatoarele: realizarea pozelor, afișarea, salvarea sau ștergerea lor, zoom in/out, rotirea lor, modificarea contrastului sau a luminozității și decuparea pozelor(folosind mouseul). Pentru ca această aplicație să fie mai ușor de folosit și să nu fie necesară prezența permanentă la laptop sau PC în timpul predării orelor a utilizatorului, în completarea ei s-a construit și o aplicație mobilă care integrează cele mai importante comenzi și funcționalități ale aplicației. Aplicația mobilă va comunica cu aplicația desktop prin Bluetooth.

Aplicația mobilă permite controlul direct și de la distanță a camerei foto, calibrarea, procesarea imaginilor și afișarea acestora în timp real pe proiector. În timpul prezentării lecțiilor sau proiectelor, utilizatorul poate să deruleze pozele făcute sau importate, astfel folosind aplicația ca o diaporamă. De asemenea utilizatorul poate comuta ușor între afișarea fișierelor ce țin de curs și pozele realizate cu ajutorul aplicației.

Prin urmare acest proiect rezolvă unele din problemele prezentate anterior și aduce unele plusuri care nu țin neaparat de aceste probleme dar care sunt importante pentru ca aplicația să fie usor de folosit și să aibă funcționalități bune asemănătoare unui editor de poze. Astfel profesorul care o folosește poate ca atunci când nu mai are loc să scrie pe tablă, să fotografieze tabla și să revină oricând la pozele anterioare. De altfel, proiectul fiind realizat în special pentru Facultatea de Arhitectură și Urbanism aplicația va fi utilă în a afișa mai multe variante de desen(figuri și planuri arhitecturale) suprapuse și acestea vor putea fi rotite cu un anumit număr de grade. Ele vor putea fi și micșorate sau mărite. Folosirea acestei aplicații nu necesită un lucru de mare precizie tocmai din considerentul de a nu consuma mult timp în timpul utilizării ei, fapt ce ar lua din timpul predării.[1]

Aportul personal la acest proiect îl reprezintă realizarea de la zero aplicațiilor necesare funcționării acestui sistem, ceea fapt ce implică mai multe etape din procesul de dezvoltare software. Proiectul va fi parte a articolului științific menționat la [1], el fiind partea tehnică a acestui articol. Pentru problematica prezentată în [1] și rezumată mai sus acest proiect încearcă să aducă o rezolvare.

## 1.3 Structura Lucrării

Această lucrare este alcătuită din 7 capitole. Acestea sunt :

* Capitolul 1. Introducere: cuprinde câteva cuvinte despre domeniul, utilitatea temei, motivația și contextul proiectului.
* Capitolul 2. Tehnologii folosite: documentarea cunoștiințelor necesare realizării proiectului. Acestea reprezintă fundamentarea teoretică a lucrării.
* Capitolul 3. Specificațiile aplicației: Descrierea interfețelor dintre aplicația software și mediu(sistemele cu care interacționează aplicația).
* Capitolul 4. Proiectarea aplicației: Descrierea arhitecturii și a componentelor programului
* Capitoul 5. Utilizarea sistemului : Prezentarea modului de utilizare a aplicației
* Capitolul 6.Conzluzii: Ce s-a realizat în acest proiect, problemele întâmpinate și anumite direcții de dezvoltare
* Capitolul 7.Bibliografie

# Capitolul 2.Tehnologii folosite

În acest capitol vor fi prezentate tehnologiile folosite pentru realizarea acestui proiect, ele fiind uneltele de bază în dezvoltarea oricărui sistem software. Vor fi abordate tehnologiile, criteriile și metodele utilizate pentru dezvoltarea celor 3 subsisteme ale acestui proiect, și anume: Aplicația Desktop, Aplicatia Mobilă și Aplicația pe microcontrolerul Arduino. Toate aceste 3 aplicații au fost construite folosind tehnologii diferite, limbaje de programare diferite și au fost interconectate prin anumite protocoale.

## 2.1 Platforma Microsoft .NET Framework

Pentru a construi aplicația de bază a acestui proiect, și anume aplicația desktop, s-a utilizat platforma de dezvoltare .NET creată de Microsoft. Aplicația a fost dezvoltată pentru sistemul de operare Windows și a fost construită cu ajutorul tehnologiilor .NET precum: Windows Forms, C# și Windows Presentation Foundation (WPF) și a bibliotecilor Aforge.Net și EmguCV. Mediul de dezvoltare folosit a fost Microsoft Visual Studio .NET 2017.

Platforma .NET este o platformă de dezvoltare open source de uz general, folosită la dezvoltarea aplicațiilor pe sistemele de operare: Windows, Linux, iOS și Android. Implementările platformei .NET depind de sistemele de operare menționate anterior și de tipul aplicației dezvoltate(server, desktop, mobilă ). .Net este o mulțime de framework-uri care au o infrastuctură comună cu mici diferențe între acestea.[3]. Aceasta a apărut în anul 2002 și a devenit rapid o parte de bază în dezvoltarea software din prezent. Introducerea lui a avut ca scop a oferi programatorilor un model de programare mai puternic, mai flexibil și mai simplu față de ceea ce oferea predecesorul acestuia, COM. Cele mai importante funcționalități și caracteristici oferite de .NET sunt:

* Oferă suport pentru mai multe limbaje de programare: C#, Visual Basic, F# , C, C ++ și asa mai departe.
* Biblioteci de bază cuprinzătoare(*.NET Framework Class Library)*. Această bibliotecă cuprinde mii de tipuri predefinite care premit programatorului să construiască: alte biblioteci, aplicații de tip consolă, aplicații grafice destkop și aplicații web la nivel de întreprinderi.
* Un nucleu de runtime comun tuturor limbajelor .NET. Acesta este *Common Language Runtime* sau CLR și are rolul de a localiza, încărca și a gestiona obiecte .NET în numele programatorului. De asemenea CLR se ocupă de gestionarea firelor de execuție și de verificări de securitate.
* Integrarea limbajelor: .NET suportă moștenirea, debuggingul tratarea erorilor cross-language. [4]



Fig 2.1 .Implementări ale .NET[img 1]

**.NET Core** este o versiune .NET care este cross-platform, adică ea poate să ruleze pe orice sistem de operare(Windows, Linux, MacOS).

**Xamarin** este o platformă cu ajutorul căreia pot fi dezvoltate aplicații native Android, iOS, MacOS și Windows folosind limbaje .NET. Această platformă este bazată pe Mono - o comunitate susținută de de Xamarin/Microsoft.

.**NET Framework** dupa cum se vede în Figura 2.1, e o platformă special construită pentru dezvoltarea aplicațiilor specifice familiei sistemelor de operare Windows. .NET dispune de funcționalități specifice Windows dar are și biblioteci care extind aceste funcționalități. Multe dintre aceste biblioteci sunt folosite și pentru dezvoltarea aplicației desktop din acest proiect. [2].



Fig 2.2 Arhitectura .NET [img2]

Printre entitățile de bază în .NET, după cum se poate vedea și în Figura 2.2 se numără: CLR: *Common Language Runtime,* CTS: *Common Type System* și CLS: *Common Language Specification.* CTS definește toate tipurile de date posibile, modul în care entitățile interacționează între ele și în care un tip poate moșteni un alt tip de dată. CLS este un set de specificații pe care un limbaj de programare trebuie să le aibă pentru a fi un limbaj .NET. CLR este o mașină virtuală prin care sunt gestionate programele la rulare în ceea ce privește necesitățile acestora. Astfel, programatorul nu trebuie să țină cont de tipul procesorului pe care rulează aplicația lui. CLR este primul nivel din .NET Framework. În Figura 2.3 este descris procesul de rulare a unui program scris în unul din limbajele .NET. [2] [4]

CLR execută un cod octet numit *Common Intermediate* Language(CIL) sau *Microsoft Intermediate Language* (MISL). Acest cod nu poate să fie executat direct de procesor, codul nativ fiind creat de către Compilatorul Just-in-Time(JIT Compiler) din codul CIL. [2]



Fig 2.3 Diagrama CLR[img3]

Ca și IDE s-a folosit **Microsoft Visual Studio 2017.** Acesta este un IDE pentru dezvoltarea aplicațiilor desktop, consolă, web, mobile etc. Acesta folosește platforme de dezvoltare Microsoft precum Windows Forms, WPF sau Windows Store, suportând 36 de limbaje de programare.[5]

**2.1.1 Limbajul C#**

După cum s-a menționat și mai sus **limbajul C#** și platforma .NET au fost aduse în folosință în anul 2002 cu scopul de a oferi programatorilor ceva mai mult decât oferea COM. Limbajul de programare C# are o sintaxă de bază care seamănă foarte mult cu *Java*,dar C# face parte din familia limbajelor C( C, Objective C ,C ++). C# este un limbaj orientat pe obiecte la fel ca și Java, amândou[ având sintaxă bazată pe C, acestea evoluând din C++. C# , la fel ca și C++ permite supraîncărcarea operatorilor, crearea structurilor de date, enumerărilor și a funcțiilor de tip callback(cu delegați) [6][7] .

În plus, C# are unele funcționalități speciale regăsite în diferite limbaje funcționale (LISP sau Haskell) cum ar fi : expresiile lambda și tipurile anonime. Odată cu apariția LINQ (*Language Integrated Query*)*,* C# devine tot mai mult un limbaj unic în universul programării. Din cauză că C# este un hibrid a mai multor limbaje, rezultatul este un limbaj curat din punct de vedere sintactic(ca și Java), simplu ca si Visual Basic și puternic și flexibil ca și C++. O caracteristică importantă care reiese din Fig 2.3 este că C# produce cod care se poate executa doar în runtime(la momentul rulării).[6][7]

Câteva dintre caracteristicile mai importante ale limbajului C# sunt:

* Nu este necesară folosirea pointerilor.
* Gestionarea memoriei se face automat cu ajutorul asa numitului colector de gunoaie(*Garbage collector*).
* C# oferă suport pentru programarea bazată pe atribute. Sunt permise adnotări ale tipurilor și ale membrilor acestora.
* Operatorul lambda ( => ), care simplifică mai mult lucrul cu tipurile *delegate* din .NET Framework.
* Suport pentru căutarea dinamică a membrilor la rulare(runtime) folosind cuvântul cheie *dynamic.*
* Permite programarea asincronă și cu fire de execuție care au fost simplificate mult prin apariția cuvintelor cheie : *async* și *await.*
* Posibilitatea de a construi tipuri generice și membrii generici. Folosind membrii generici, codul scris poate să fie type-safe, care definește numeroși substituenți specificați atunci când se interacționează cu tipul generic.
* Un nou fel de inițializare a obiectelor, care permite setarea valorii proprietățiilor la momentul creării acestora.
* Parametrii opționali în cadrul metodelor.[7]

În Martie 2017 a apărut curenta versiune majoră de C# și anume C# 7. Aceasta a adus noi funcționalități pentru a eficientiza baza de cod și a mai adus tuplii, variabilele și parametrii *ref.* Alte funcționalități :

* Existența tupliilor(tipuri de categorie ușoară fără nume) precum și a variabilelor locale și returnate *ref*(returnarea referinței unei valori în loc de valoarea în sine ).
* Tipurile returnate de tip async generalizate
* Un nou cuvânt cheie *default* care permite inițializarea oricarui tip

Posibilitatea ca metoda *main* a unui program să fie *async* [7]

În data de 7 Mai 2018, C# a ajuns la vresiuna stabilă 7.3. În acest proiect a fost folosită versiunea 4.7.1 a platformei .NET care a fost instalată odată cu instalarea IDE-ului Visual Studio 2017(versiunea 15.4.4), iar versiunea C# folosită a fost 7.2. [3]

**2.1.2 Multithreading, Programarea Concurentă și Async**

Aplicațiile care funcționează bine sunt cele care au un timp de răspuns foarte bun și în cadrul cărora atunci când se pornește un task, acesta nu împiedică alte părți din program să funcționeze așa cum trebuie. Realizarea acestui lucru necesită operații mai complexe în C++ dar platforma .NET a adus cu ea căi de realizare a multithreading-ului mult mai ușoare. Un fir de execuție este definit ca o cale de execuție într-o aplicație. Prin crearea firelor de execuție adiționale se pot construi aplicații mult mai receptive. Spațiul de nume *System.Threading* a apărut odată cu .NET 1.0 și oferă suport pentru a construi aplicații cu fire de exectuție[7]

În acest proiect recepționarea datelor de la portul serial(în cadrul aplicației desktop) s-a realizat în mod asincron și a fost necesar un alt fir de execuție decât cel al interfeței grafice. Pentru a transmite datele de la firul nou creat la firul interfeței grafice este nevoie de o funcție specială de tip **delegate.** Un delegat .NET este practic un pointer la o funcție type-safe. Delegații au rolul de a transmite metode ca și parametrii pentru alte metode. Delegatului îi pot fi transmise metode care au aceeași semnătură cu a sa(semnătura include și valoarea returnată). Delegații sunt ideali pentru metodele de tip *callback*. [7][2].

## 2.2 Programarea orientată pe eveniment

Programarea orientată pe eveniment este o paradigmă de programare din programarea calculatoarelor în care cursul unui program este determinat de anumite evenimente precum primirea informațiilor de la senzori, clickul unui mouse, o tastă apăsată, trecerea unui anumit timp etc.[9] Ceea ce diferențiază programarea orientată pe eveniment de alte paradigme de programare este că această paradigmă este un mod de a gândi problemele și soluțiile lor. Ea furnizează abstractizări. Limbajele sunt cunoscute ca și limbaje funcționale și orientate pe obiecte pentru că ele atribuie abstracțiuni unor structuri de program. La fel se întâmplă și cu programarea bazată pe eveniment, modelul evenimentelor fiind abstractizarea principală.[9].

La baza programării orientată pe eveniment stă conceptul de eveniment. Trei tipuri de obiecte computaționale sunt associate fiecărui eveniment : sursa evenimentului(*event source*), obiectul eveniment(*event object*) și unul sau mai multe proceduri eveniment(*event handlers*).[8]

În realizarea acestui proiect, pentru aplicația Desktop și cea mobilă în mare parte s-a folosit paradigma programării orientate pe eveniment. Elementele cele mai importante din această paradigmă sunt prezentate mai jos:

* *Sursele evenimentului (Event Sources)*.

Acestea reprezintă originea evenimentului. O sursă declanșează un eveniment atunci când crează un obiect eveniment și îl pregătește pentru procedurile de evenimente. Sursele de de evenimente pot să fie obiecte de tip software, hardware sau firmware.

* *Obiectele sursă*(*Event Objects)*

Un eveniment obiect încapsulează datele importante cu privire la un eveniment. Clickul unui mouse, de exemplu, este printre cele mai des întâlnite evenimente. Datele corespunzătoare fiecărui click sunt: coordonatele ecranului sau ferestrei curente și butonul care a fost apăsat. Orice procedură eveniment care “ascultă” aceste clickuri va avea nevoie de aceste informații așa că orice procedură, indiferent de limbaj va avea nevoie de ele.

* *Procedurile Eveniment(Event Handlers)*

Procedurile eveniment răspund la evenimente prin executarea acțiunilor impuse de programator. Aceste proceduri sunt factorul care unește diferite obiecte pentru a întocmi un program funcțional. Procedurile sunt asociate cu sursele lor(event sources), apoi asteaptă ca sursa să declanșeze evenimentul



Fig 2.5 Cursul unui program orientat pe evenimente[img4]



Fig 2.6 Modelul de procesare al evenimentelor

pentru programarea orientata pe obiecte[img5]

Intrfețele grafice(GUI) sunt o bună cale de a ilustra baza programării pe eveniment. Partea fundamentală a programării orientate pe obiecte sunt clasele și obiectele. Același lucru se aplică și pentru cea orientată pe eveniment. Figura 2.5 și Fig 2.6 arată relația dintre diferitele părți prezentate mai sus.[8]

## 2.3 Biblioteci speciale: AForge, OpenCV

Pentru lucrul cu camera foto/web și cu imaginile primite de la aceasta s-a optat pentru folosirea framework-ului **AForge .NET** și **Emgu CV**( wrapper pentru **Open CV**).

AForge .NET este un framework scris în C#, open-source, creat pentru dezvoltatori și cercetători din domeniile: Inteligență Artificială, procesare de imagini, rețele neuronale, computer vision. De asemena mai poate fi utilizat în logică fuzzy, machine learning și robotică. În acest proiect acest framework a fost folosit pentru a detecta camerele web conectate la PC, pentru a prelua cadrele și xtrage un frame la un mompentru a realiza poze(a eentdat). A mai fost folosit și pentru a edita anumite imagini.

Această platformă este alcătuită din multe librării: AForge. Imaging, AForge.Vision, AForge.Neuro, AForge.Fuzzy. Pentru proiect au fost folosite AForge.Video și AForge.Imaging. Pentru a capta informații de la camera web s-au folosit clasele *FilterInfoCollection* și *VideoCaptureDevice* și evenimentul de *NewFrame*. [10]

**Emgu CV** este o înfășurătoare .NET a frameworkului **OpenCV**(folosit la procesarea imaginilor, C++)care permite utilizarea funcțiilor OpenCV în cadrul plaformei .NET. Emgu CV poate fi compilat în Visual Studio, Xamarin, Unity și poate rula pe Windows, Android, MacOS etc. EmguCV este scris în totalitate în C#. În Figura 2.7 se poate vedea arhitectura OpenCV și înfășurătoarea acestuia. Fișierele cu extensia .dll(dynamic-link library) sunt fișierele în care sun integrate aceste clase și funcționalități provenite din OpenCV.[11]



Fig 2.7 Arhitectura OpenCV și wrapper-ul EmguCV[img6]

## 2.4 Microcontrolerul Arduino Uno. Modulul Bluetooth

Unul din subsistemele care este parte componentă a acestui proiect/sistem, este subsistemul format din microcontrolerul **Arduino Uno** la care este conectat modulul Bluetooth HC-05. Alături de acestea este inclusă aplicația dezvoltată pentru Arduino în mediul de dezvoltare open-source Arduino Software IDE. Aplicația preia date de la aplicația mobilă prin Bluetooth și le transmite aplicației desktop prin portul serial. Față de celelalte părți din aplicație acest subsistem implică o parte software dar și o parte hardware.

* **Microcontrolerul Arduino Uno**

Acest microcontroler(Fig 2.8) are la bază procesorul ATmega328. Are 14 pini digitali de intrare/ieșire, 6 intrări anlogice și un oscilator ceramic rezonator de 16MHz. Din cei 14

pini digitali, 6 pot fi folosiți ca și ieșiri PWM. Ca și conectivitate, Arduino mai are conexiune USB(și pe post de alimentare), alimentare AC-to-DC sau baterie. Numele “Uno” are rolul de a marca prima versiune de Arduino, versiunea 1.0 [12]



Fig 2.8 Microcontrolerul Arduino Uno[img7]

Procesorul ATmega328 creat de Atmel din familia megaAVR, dispune de o memorie flash de 32 Kb, 1 Kb EEPROM, 2 Kb SRAM și 32 de regiștrii.

**Intrari și ieșiri:**

* Seriale : pinul 0 și 1, aceștia fiind RX și TX. La acești pini va fi conectat și modulul Bluetooth.
* Întreruperi externe: pinii 2 și 3. Pot să fie setați să declanșeze întreruperi la front crescător sau front descrescator.
* PWM : pinii 3,5,6,9,10,11 – PWM pe 8 biți
* SPI(Serial Peipheral Interface) : 10(SS) ,11(MOSI) ,12(MISO), 13(SCK).

Arduino poate fi programat prin software-ul Arduino. ATmega328 vine încărcat cu un *bootloader* care permite încărcarea de nou cod. Acesta comunică prin intermediul protocolului STK500. Arduino are funcționalitatea de a fi resetat de la buton dar și atunci când este rulat pe el software de pe un calculator.[12]

* **Limbajul C++**

Programul care a fost scris pentru aplicația Arduino a fost scris în C++. Bjarne Stroupstrup de la AT&T Bell Labs a creeat C++ la mijlocul anilor 80 ca și o extensie a limbajului C. Limbajul C era folosit pentru mai mult pentru software la nivelul sistemului și pentru sisteme embedded. C++ a venit în plus cu funcționalități specifice limbajelor orientate pe obiecte. Programele în C++ au nevoie a fi compilate. Un compilator traduce codul țintă. Acest cod poate fi cod mașină pentru o anumită platformă sau în alt limbaj(primele compilatoare C++ traduceau codul în C). Setul întreg de tooluri pentru C++ include:

* *Preprocesor:* adaugă și modifică conținutul fișierului sursă înainte de procesarea codului de către compilator.
* *Compilatorul*: traduce codul sursă în cod mașină.
* *Linker :* combină codul-mașină generat cu biblioteca precompilată sau cod compilat din alte surse pentru a face un program executabil complet.[18]

Un program scris corect în C++ are o anumită structură. Sintaxa trebuie să fie corectă sau compilatorul va genera mesaje de eroare și nu va produce cod mașină executabil. Un fișier sursă pentru C++ are extensia .cpp. Un fișier sursă începe de obicei cu directive ce țin de preprocesare, iar toate directivele de preprocesare din codul sursă C++ încep cu simbolul „#”. Directiva **#include** specifică un fișier numit *header* care conține specificațiile pentru codul din bibliotecă. Bibliotecile folosite în acest proiect sunt : “Arduino.h” (pentru a avea acces la funcții specifice microcontrolerului Arduino), “SoftwareSerial.h”(cuprind funcțiile ce permit lucrul cu portul serial – USB).[18]

* **Protocolul USB**

În acest proiect, legătura dintre subsistemul aplicației pe microcontrolerul Arduino și cel al aplicației desktop este realizată cu ajutorul portului serial USB. De asemenea, camerele foto/web sunt conectate tot prin USB la calculator.

USB(Universal Serial Bus) este soluția ideală pentru a conecta la calculator diferite dispozitive externe acestuia. Printre multele calități ale interfeței se numără: este ușor de folosit, este rapidă, versatilă(pot fi conectate multe feluri de periferice la ea), necostisitoare, apariția rară erorilor și nu consumă multă energie. De asemenea, este suportată de Windows și de alte sisteme de operare. Fiecare calculator modern(PC sau Macintosh) are porturi USB la care pot fi conectate periferice standard : tastaturi, mouse-uri, scannere, camere web/foto, imprimante, dar și alte părți hardware pentru diferite scopuri.

Portul USB are format serial asincron și la el pot fi conectate maxim 127 de dispozitive. Distanța maxima la care pot fi legate dispozitivele este de aproximativ 5 metri. Aceasta se întinde până la 30 de metri cu 5 hub-uri. Când utilizatorul conectează cablul USB la calculator, sistemul Windows detectează dispozitivul și încarcă driver-ul corspunzător. Un dispozitiv periferic USB poate fi conectat și deconectat oricând fără a-i cauza defecțiuni.

Microcontrolerul Arduino are incorporat un port USB pentru un cablu USB- TypeB care se conecteză la PC. Prin acest cablu Arduino nu doar că transmite date spre PC dar el este și alimentat la +5V nemaifiind nevoie de alimentare de la priză sau baterie. Într-o aplicație de genul acestui proiect, viteza transmiterii datelor este foarte importantă pentru a avea un sistem care tinde a fi un sistem în timp real.[20] USB 2.0 a apărut în 2001( fiind folosit și pentru acest proiect) și suportă următoarele lățimi de bandă:

-Low-speed – 1.5Mb/sec: pentru dispozitive precum mouse-uri, tastatură.

-Full-speed – 12Mb/sec

-High-speed – 480Mb/sec pentru dispozitive precum stickuri, harduri externe, căști, imprimante. USB 3.0 a apărut în 2011, incorporând funcționalitățile lui USB 2.0 pe lângă magistrala SuperSpeed. Acesta are viteza de transfer a datelor de aproximativ 625Mb/s, iar USB 3.1(2014 – SuperSpeed+), ajunge la 2.5Gb/s.[21]

* **Modulul Bluetooth HC-05**

Pentru a transmite date de la aplicația mobilă la apilcația desktop s-a folosit acest modul bluetooth conectat la Arduino Uno, ca și un intermediar. A fost aleasă calea Bluetooth pentru că majoritatea telefoanelor smart-phone moderne sunt dotate cu această tehnologie.

**Protocolul Bluetooth**  ****

Tehnologia wireless Bluetooh a fost dezvoltată in 1994 la Ericsson în Suedia. El a fost dezvoltat cu scopul de a elimina nevoia de conexiuni prin cablu la comunicarea între diferite dispozitive. În 1998 a fost formată SIG – Bluetooth Special Interest Group care a creeat Specificațiile Bluetooth IEEE 802.15. Acestea au standardizat dezvoltarea dispozitivelor care au bluetooth astfel încât acestea, chiar dacă sunt produse de producători diferiți să poată comunica între ele. Dispozitivele Bluetooth funcționează fără restricții pe banda de 2.400 GHz – 2.483 Ghz. Această bandă se numește ISM(Industrial,Science,Medical band). Dispozitivele Bluetooth folosesc o tehnică numită *frequency hopping* pentru a diviza datele transmise în bucăți numite pachete. Transmițătorul și receptorul schimbă un pachet de date între ele la o frecvență și apoi se sare la altă frecvență pentru a transmite un alt pachet. [13]

Securitatea Bluetooth este complexă dar este transparentă și usor de implementat pentru utilizatori. Sunt folosite 3 tipuri de securitate:

-*autentificarea* : se verifică identitatea dispozitivului cu care se încearcă conectarea.

-*autorizarea* : după realizarea autentificării unul din dispozitivele Bluetooth garantează/autorizează un alt dispozitiv Bluetooth la un anume serviciu.

-*encriptarea* : transformă datele într-un format care poate fi citit doar de un dispozitiv care are aceeași cheie de encriptare.

Există o procedură de a creea o legătură între două dispozitive numită asociere (*pairing*)*,* care il scutește pe utilizator să refacă conexiunile de la 0 de fiecare dată când conectează două dispozitive Bluetooth[13]. Pentru acest proiect s-a realizat o asociere între telefon și modulul Bluetooh și acestea rămân asociate chiar dacă unul din ele se închide.



Fig 2.9 Modulul Bluetooth HC-05[img8]

HC-05(Fig. 2.9) este un modul Bluetooth ușor de folosit, proiectat pentru a putea fi configurat și conectat de către utilizator folosind anumite microcontrolere. El poate fi conectat și la un PC. Este un modul Bluetooth SPP(Serial Port Protocol) făcut pentru conexiune wireless serială. Față de modulul HC-06, HC-05 este mai versatil, el putând fi folosit atât ca și master, dar și ca și slave.

În Fig 2.9 se poate observa că HC-05 are 6 pini. Aceștia sunt:

**-** *VCC* : alimentarea la +5V.

*- GND* : conectarea la masă (Ground).

*- TDX* : UART\_TDX, pinul de transmitere serială Bluetooth. Se conectează la pinul RDX(Receive) al microcontrolerului sau al altor dispozitive.

-*RDX* : UART\_RDX, pinul de recepție serială Bluetooth a datelor. Se conectează la pinul TDX(Send) al microcontrolerului sau al altor dispozitive.

-*STATE* : Acest pin este conectat la LED-ul plăcuței și poate fi folosit ca și feedback pentru a verifica dacă Bluetooth-ul funcționează bine.

-*EN* : Enable/Key – Pinul este folosit pentru a comuta între Data Mode și comanda AT(attention). Modul implicit este Data mode.

Plăcuța mai are pe ea un LED care indică starea modulului:

- LED-ul pâlpâie o dată la 2 secunde. Modulul este în modul Comandă.

- Pâlpâiri repetate: așteptarea conexiunii în modul Data

-LED-ul pâlpâie de două ori într-o secundă. Conexiune realizată cu succes în modul Data.

Modulul dispune și de un buton care este folosit pentru a controla manual pinul Key/Eneble(*EN*) pentru a comuta între modul Data și modul Comanda. **Modul Data** este modul în care HC-05 poate să transmită și să recepționeze date de la alt modul Bluetooth. **Modul Comanda(AT)** este modul în care setările implicite ale modulului pot fi configurate. [14] [15]

## 2.5 Android, Java și Android Studio

**2.5.1 Android**

După cum s-a menționat în capitolele de mai sus, o parte componentă a sistemul acestui proiect este aplicația mobilă prin care se transmit date de la smart-phone la modulul Bluetooth conectat la Arduino. Tehnologiile care s-au folosit pentru realizarea acestei aplicații mobile au fost *Android, limbajul Java* și *Mediul Android Studio.*

Android este un sistem de operare dezvoltat de Google pentru în special dispozitive cu ecran tactil precum telefoane mobile(smartphone) și tablete. Are la bază un Kernel de Linux. Interfața cu utilizatorul este bazată pe realizarea unor mișcări și gesturi care corespund cât de cât cu acțiuni din lumea reală(glisare, bătaie înceată, ciupit etc.). În plus Android mai este folosit și la televizoare, ceasuri, mașini, notebook-uri, console de jocuri, camere digitale și alte dispozitive. În acest proiect s-a ales lucrul cu sistemul de operare Android pentru aplicația mobilă pentru că, după cum se menționează și în [16], Android este cel mai popular sistem de operare pentru telefoane din industria mobilă. Pe aproximativ 2 miliarde de dispozitive rulează Android.[16].

Android este un sistem open-source. Google a făcut codul pentru părțile low-level și pentru cele necesare să alimenteze dispozitivul electronic, dar a dat gratis Android pentru utilizatorii care doresc să scrie cod și să în continuare sistemul acesta. Există un framework întreg pentru aplicații pentru ca aplicații de terță parte să fie dezvoltate și instalate. Acest lucru reprezintă *Android Open Source Project.*

Pentru dezvoltarea aplicațiilor Android s-a folosit mediul de dezvoltare*Android Studio*, tool introdus de Google. Acesta este IDE-ul oficial pentru programarea aplicațiilor Android. Acesta poate fi descărcat de pe internet și include tot ce trebuie pentru a putea începe crearea aplicațiilor. Sunt incluse: *kit-ul de dezvoltare software(****SDK)*** cu toate *bibliotecile Android necesare* precum și *simulatorul Android*. Acest simulator permite dezvoltatorului să testeze aplicația fără să aibe nevoie de un dispozitiv real cu sistemul Android. Google lansează aproape în fiecare an o nouă versiune de Android pentru a actualiza sistemul de operare și a repara erori. Deci fiecare versiune de Android are propriul ei *SDK*  pentru a putea construi aplicații care au ultimele funcționalități Android adăugate.[16]

**2.5.2 Java**

În primii ani din anii 90 un grup de ingineri de la Sun Microsystems au avut viziunea că viitorul în tehnologie va fi legătura între dispozitive client digitale și calculatoare. Conduși de James Gosling echipa a dezvoltat un limbaj nou de programare revoluționar: Java (inițial Oak). Astăzi, Java stă în spatele multor aplicații și dispozitive care “alimentează” viața și tehnologia din viata de zi cu zi. De la telefoane mobile la alte dispositive de mână, jocuri, sisteme de navigație, soluții business, Java este pretutindeni. Este unul din cele mai populare limbaje de programare din industria IT.

Pentru a scrie un program în Java este nevoie de un software pentru a scrie codul. Acesta poate fi un compilator Java sau un IDE(Integrated Development Environement) precum: Eclipse, InteliJ Idea, Android Studio și altele. Ele sunt numite Javac. În acest proiect s-au folosit *Android Studio* și *compilatorul Java.* Codul sursă Java scris în Android Studio e transmis într-un fișier *project.class.* Acest fișier va fi transferat JRE-ului(*Java Runtime Environement*). JRE are o parte numită Class Loader care primește acest fisier .class și motorul de execuție (*Execution Engine*), care este parte din JRE îl execută. Aceste acțiuni se pot vedea în figura Fig.2.10



Fig 2.10 Fluxul de lucru a unui program Java[img9]

O parte principală în a crea și a exectua un program scris în Java îl reprezintă JDK(*Java Development Kit*). Acest software include partea care rulează și scrie codul apoi trimite rezultatul spre sistemul de operare. JDK include o mașină virtuală Java(*JVM*). JVM permite calculatorului să ruleze programe Java, dar și alte programe în alte limbaje care sunt compilate în codul masină Java. Acest cod masină(din fișierul .class) este rezultatul compilării cu compilatorul Javac. Ceea ce reprezintă un mare avantaj a limbajului Java și al *JVM*  este următorul: chiar dacă fiecare sistem de operare are o JVM diferită, rezultatul produs după execuția codului mașină este același în cadrul tutror sistemelor de operare.**[**17]

## 2.6 Testarea

Pentru ca un produs software să fie de succes, testarea software are un rol esențial în acest lucru. În majoritatea cazurilor, în dezvoltarea produselor software în producție, există echipe special concepute pentru a realiza testarea produselor. Cu cât produsul este mai complex, cu cât are mai multe componente și legături mai multe între acestea, cu cât codul lui este mai complex, cu atât testarea trebuie să fie mai solidă și de calitate. Acest proiect fiind dezvoltat doar de o persoană și fiind un proiect mai mic, față de cele existente pe piață, testarea lui nu a constituit un proces foarte elaborat. Cele mai multe teste au fost cele unitare, testele de dezvoltator(testele manuale) și testele de sistem.

**Testarea Manuală**

Testarea manuală este cea mai bună cale de a găsi erori în cod nou scris. Acestea pot fi executate pe loc după fiecare nouă funcționalitate adăugată. Dar pe măsură ce aplicația devine mai complexă și are tot mai multe funcționalități, testarea manuală devine tot mai costisitoare din punct de vedere al timpului și al efortului depus. De asemenea, rerularea testelor este costisitoare. În mod normal se testează doar ultimele funcționalități adăugate, testele manuale mai vechi neputând fi păstrate. Unul din scopurile testelor unitare este acela de a proteja împotriva regresiilor și anume a ținea codul stabil în timp ce se lucrează la el. Ele nu sunt ca și testele de sistem. Testele unitare rulează pe un server și nu sunt legate neaparat de cerințe. [22]

Testarea unitară este o modalitate de testare prin care scriu un set de teste prin care se testează anumite unități de cod și anumite module din acestea. În literatura software, o unitate reprezintă cea mai mică colecție/parte de cod care poate fi testată. Această unitate poate fi văzută ca și o procedură sau o funcție din programarea funcțională sau o clasă sau interfață în programarea orientată pe obiecte. Testarea unitară e unul din nivelele testare ca ajută la a completa testarea unui sistem. Ea este văzută ca și o *testare “white box”.* Se bazează mai mult pe testarea codului ca și implementare nu neaparat ținând cont de cerințele după care codul a fost scris. Testarea unitară este importantă pentru că este foarte ineficient ca un sistem să fie testat doar ca o cutie neagră (*testare “black box”*). Acest tip de testare vizează mai mult comportamentul sistemului și nu verificarea codului. [23]

## 2.7 Protocolul SMTP

SMTP – Simple Mail Transfer Protocol(Protocolul simplu de transfer de postă) este un protocol prin care se realizează transferul unor mesaje - email de la expeditor la destinatar de către anumiți agenți de transfer. SMTP permite doar trimiterea emailurilor nu primirea acestora. Acesta este cel mai simplu mod de transmitere. În Internet, un email este trimis prin faptul că expeditorul realizează o conexiune de tip TCP la portul 25 (portul SMTP) al destinatarului. La acest port “ascultă” un server de mail care cunoaște protocolul SMTP. Acest server(“yahoo”, ”gmail”, ”hotmail” și altele) acceptă conexiunile, relizează niște verificări de securitate și acceptă mesajul care trebuie transmis. Dacă mesajul nu poate fi distribuit, o eroare care conține prima parte a mesajului va fi returnată. După stabilirea conexiunii TCP la portul 25, expeditorul(clientul) va astepta serverul să “vorbească” primul. Serverul trimite o linie prin care își spune identitatea și dacă este pregatit sau nu să primească un mail. Daca acceptă să trimită, clientul spune de la cine vine mailul și unde trebuie să ajungă. Dacă destinatarul există, serverul dă undă verde pentru expeditor.

Un dialog între server și client începe cu comanda *HELLO*. Prin comanda *RCPT*, clientul îi specifică serverului cine va fi recipientul sau recipienții mesajului. SMTP este limitat în anumite aspecte, unul din ele fiind că nu include autentificare. Deci comanda *MAIL FROM* ar putea să dea orice adresă a unui expeditor fiind folositor pentru a trimite spam. Prin comanda *DATA* se va specifica conținutul mesajului iar prin comanda *QUIT* se termină sesiunea. Toate aceste comenzi trebuie să fie cunoscute de un server SMTP. Portul recomandat pentru a fi folosit este 587 pentru că mai toate serverele suportă acest port. [25]

# Capitolul 3.Specificațiile aplicației

Acest capitol urmărește modul în care au fost aplicate fundamentele teoretice prezentate în capitolul 2, în vederea realizării acestui proiect. Se vor urmări modalitățile practice de realizare a modelelor software și sistemul fizic. Se vor descrie interfețele dintre aplicația software și mediu(sistemele cu care aceasta comunică). Se va prezenta interacțiunea cu utilizatorul sau utilizatorii aplicației și sistemele cu care interacționează programul. Vor fi prezentate și perifericele calculatorului pe care rulează aplicația(care fac sau nu parte din sistemul aplicației).

## 3.1 Schema-bloc a sistemului.

**Descrierea aplicației**

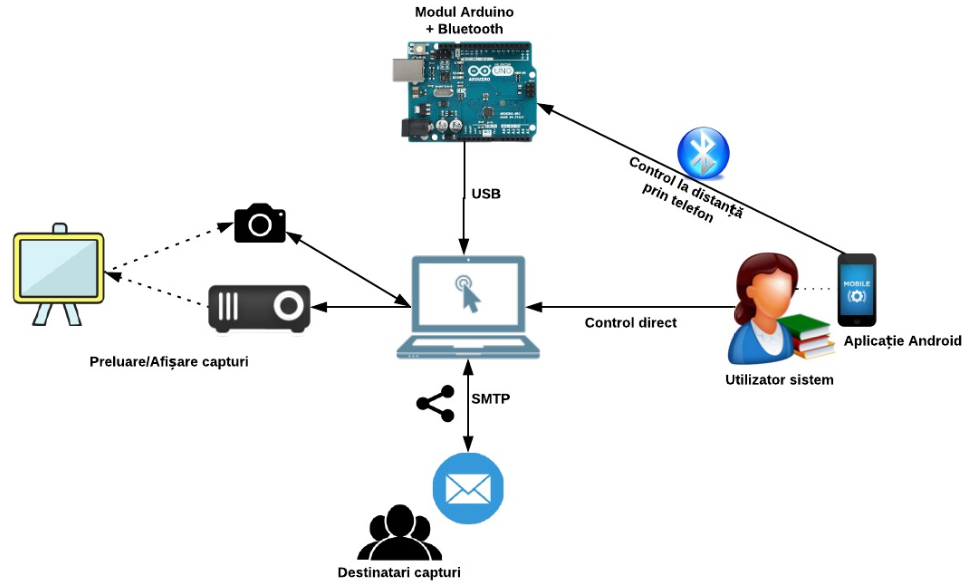


Fig 3.1 Schema-bloc a sistemului – Arhitectura

În Figura 3.1 este prezentată schema-bloc a sistemului care stă la baza acestui proiect. În ea sunt descrise schematic cele 3 subsisteme care construiesc întregul aplicației( aplicația mobilă Android, aplicația Desktop PC și aplicația pe microcontrolerul Arduino). În această figură se poate vedea că entitatea centrală în acest sistem este aplicația Desktop – PC, ea fiind practic nucleul acestei aplicații. Aplicația Desktop este și cea mai complexă aplicație și este legată direct sau indirect(în cazul aplicației mobile) cu celelalte subsisteme și elemente din acest proiect. Utilizatorul sistemului poate folosi acest sistem, după cum se vede în figură în mod direct și în mod indirect. În mod direct utilizatorul poate să folosească PC-ul pe care este instalată aplicația Desktop. În acest mod de folosire, celelalte două aplicații(mobilă și Arduino) pot să nu fie folosite, fără ca acest lucru să influențieze buna funcționare a aplicației în general. În mod indirect, utilizatorul poate să folosească o parte din funcționalitățile aplicației desktop pe canalul: aplicație mobilă – *bluetooth* – Arduino – *usb –* aplicație Desktop. Prin aplicația mobilă se pot trimite anumite comenzi de bază, nefiind necesară prezența permanentă la laptop/PC, acest lucru facilitând mult folosirea acestui sistem.

În partea din stânga a Figurii 3.1 se află partea de afișare a capturilor/pozelor și de realizare a lor. Ea este alcătuită dintr-un dispozitiv de proiectare a pozelor(de obicei videoproiector sau un monitor/ecran) și o cameră foto digitală/DSLR/web(web în acest caz). Aceste dispozitive de proiectare sunt interpretate de aplicație ca și ecrane secundare. Conexiunea dintre laptop/PC și aceste dispozitive poate să fie HDMI(digital), VGA(analogic), DVI sau altă conexiune. Ca și loc de afișare a pozelor, lucru care de altfel nu ține de specificațiile aplicației, este o tablă(albă, după cum s-a specificat în capitolul 1 al lucrării), dar și alte suprafețe(pereți, ecrane de proiecție etc.). Pentru a realiza pozele se folosește un aparat foto, o cameră video sau o cameră web conectată la PC.

În partea de jos în Fig. 3.1 se află reprezentată partea din aplicație care are legătură cu internetul. Aceasta este trimiterea pozelor facute prin aplicație, care au fost salvate, la anumite adrese de email(de exemplu elevilor prezenți la respectiva oră). Acest lucru se poate realiza doar de pe aplicația Desktop, iar protocolul folosit pentru transmitere este SMTP(Simple Mail Transfer Protocol).

Aplicația aceasta este concepută pentru a fi folosită la predarea cursurilor și la alte ore(semniarii, laboratoare) de către cadre didactice și elevi. Cu ajutorul ei se fac poze tablei pe care se scrie pentru a le putea salva, edita(mărire, micșorare, rotire stânga-dreapta, luminozitate, contrast, decupare) și afișa pe un ecran secundar sau ecran de proiecție. Camera cu care se iau capturile poate sa fie foto(folosindu-se capturile în direct – *tetherd shooting*), video sau webcam( capturile reprezentând cadre din filmare). Pentru a fi ușor și rapid de folosit, multe din comenzile aplicației se pot realiza din taste. Pentru funcționalități mai avansate precum decuparea pozelor, modificarea contrastului și a luminozității, salvarea și distribuirea pozelor prin Email, este nevoie de utilizarea și a mouseului și tastaturii. Dar în mare parte, funcționalitățile necesare suportului predării unui curs/laborator/seminar pot fi accesate prin butoane de la tastatură. Pentru a putea controla de la distanță acest sistem, lucru de folos de altfel, se poate conecta la PC prin USB modulul Arduino(cu Bluetooth). Arduino va prelua date prin Bluetooth de la aplicația mobilă Android. Aplicația mobilă Android integrează comenzi pentru cele mai importante funcționalități ale acestui sistem, în general cele prin care pot fi accesate și prin folosirea tastelor PC-ului.

## 3.2 Funcțiile sistemului

**Funcțiile aplicației Desktop**

Aplicația Desktop are cele mai multe și complexe funcționalități:

* **Conectarea la camera foto/video** – funcționalitate esențială pentru funcționarea acestui sistem, aplicația desktop are un algoritm de căutare a dispozitivelor (*imaging devices*) și permite utilizatorului să aleagă dispozitivul dorit. De asemenea se poate selecta dintr-o listă de rezoluții suportate de camera respectivă.
* **Stocarea pozelor –** odată ce o poză este realizată, ea poate fi salvată inițial într-o zonă temporară iar apoi pozele se pot salva local pe hard disk.
* **Proiectare și derulare poze –** pozele realizate și salvate temporar pot fi proiectate pe un ecran secundar(monitor, TV, videoproiector).
* **Manipularea pozelor pe ecranul secundar –** aplicația dispune de comenzi de la butoane pentru: a roti poza în ambele direcții, a deplasa-o în cele 4 direcții, a mări/micșora poza și pentru a afișa ecranul negru.
* **Decupare poze –** cu mouseul se pot decupa(*crop*) pozele și astfel o nouă poză este creată
* **Comunicare cu portul serial(Arduino – aplicație mobilă) –** aplicația Desktop poate primi comenzi de la aplicația mobilă. Acestea sunt primite prin portul serial(USB) și prin ele sunt accesate funcționalitățile principale ale aplicației.
* **Distribuire poze –** pozele realizate pot fi distribuite unei liste de emailuri.

**Funcțiile aplicației Arduino**

Aplicația Arduino are un rol de intermediar în acest sistem. Ea are în principal 2 funcții:

* **Comunicare Bluetooth –** preluarea informațiilor de la aplicația mobilă folosind modulul atașat și protocolul Bluetooh
* **Comunicare Seriala(USB) –** datele primite prin Bluetooth sunt transmise mai departe nemodificate spre aplicația Desktop folosind portul serial cu protocolul USB

**Funcțiile aplicației mobile**

Aplicația mobilă are funcționalități puține și destul de simpliste:

* **Conectarea cu alte dispositive Bluetooth –** telefonul mobil poate căuta alte dispozitive Bluetooh și poate forma pereche ce acestea pentru schimb de informații
* **Control asupra funcțiilor aplicației Desktop –** prin aplicația mobilă se pot trimite comenzi spre aplicația Desktop pe calea: dispozitiv Smart – Bluetooth – Arduino – USB – PC

## 3.3 Interfața cu utilizatorul

În acest paragraf vor fi prezentate principalele interfețe cu utilizatorul din aplicația Deskop și din aplicația mobilă.

**Interfața aplicației Desktop**

Aplicația desktop este împărțită în două interfețe princpale. Una este partea de afișare a pozelor(care de fapt este o fereastră care se deschide pe ecranul secundar), iar cealaltă este interfața cu care interacționează utilizatorul direct. Cu partea de afișare, utilizatorul nu are interacțiune directă, ea fiind doar rezultatul acțiunilor făcute în interfața principală sau în aplicația mobilă. În ceea ce urmează se vor prezenta interfețele(meniurile și ferestrele) din apicația Desktop.

Ca orice aplicație Desktop care se vrea a fi o aplicație decentă și care are anumită prezentare corespunzătoare, această aplicație porneste cu un ecran de prezentare(*splash screen*) – Figura 3.2 - cu numele aplicației și o bară de progres.



Fig. 3.2 Ecranul de introducere

După acest ecran inițial, se va deschide interfața principală(“Main Tab”), după cum se vede și în Figura 3.3.



Fig. 3.3 Interfața principală – *MainTab*

Aplicația Desktop are 3 file vizibile inițial(sau interfețe). Prima filă – *MainTab –* este prezentată în Figura 3.4 împreună cu ecranul secundar.



Fig. 3.4 Interfața principală completă

După cum se vede în Figura 3.3 și 3.4, prima filă are principalele funcții ale aplicației. De aici se poate selecta modul de lucru: modul captură(în care se fac capturi prin camera foto), modul afișare(prin care se caută poze în PC care să fie afișate) și modul de decupare a pozelor. Pentru o utilizare rapidă și usoară aici sunt trecute toate comenzile din taste, unele fiind disponibile și din interfață. Pozele realizate sunt afișate mai jos iar în partea dreaptă sunt două locuri pentru imaginea live de la cameră și pentru imaginea curentă. Doar imaginea curentă se poate afișa pe ecranul secundar după cum se vede și în ecranul din partea stângă din figură. În partea de jos se poate selecta numărul de grade de rotație a pozei și se afișează starea de conectare a aplicației mobile la cea Desktop.

Apoi, a doua filă este prezentată Figura 3.5 și aceasta este interfața de editare a luminozității și a contrastului. Imaginea curentă din prima interfață este prezentă și aici și utilizatorul poate modifica luminozitatea și contrastul, iar apoi poate salva sau nu aceste modificări. Dacă sunt salvate, ele sunt transferate în imaginea curentă(nu cea de pe ecranul secundar, încă).



Fig. 3.5 Interfața de editare a luminozității + contrast

Următoarea filă(din Figura 3.6) este cea în care se poate realiza conexiunea cu aplicația Android prin USB, asftel putând primi date de la aplicația mobilă. De asemenea ea mai are o parte în care se poate edita imaginea curentă în alte feluri decât cele folosite cu tehnologia WPF și Canvas. Această parte este una secundară fiind una din modalitățile în care s-a încercat modificarea poziției imaginilor, rotirea lor și micșorarea sau mărirea acestora. În interfață se pot vedea dispozitivele USB găsite și detalii despre acestea. De asemenea lângă aceasta, se pot vedea și mesajele, în timp real, care vin de la dispozitivul selectat. Dacă conexiunea s-a realizat cu succes, utilizatorul va fi anunțat, iar în *MainTab*  se va afișa, printr-un mesaj permanent, starea conexiunii.



Fig. 3.6 Interfața de conexiune cu aplicația mobilă

**Interfața aplicației mobile**

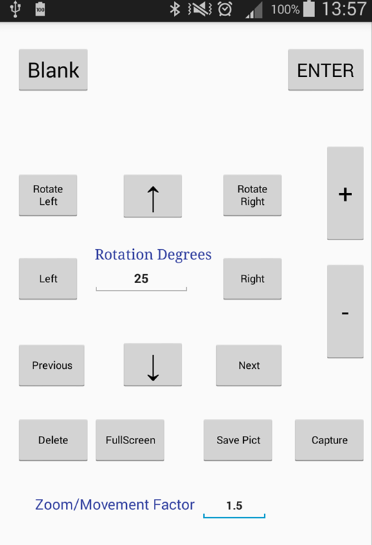
Aplicația mobilă are o interfață foarte simplă și intuitivă pentru utilizator. Ea având rolul unei telecomenzi, dispunde de multe butoane dar și de două spații text pentru introducerea numărului de grade de rotație și pentru factorul de multiplicare a zoomului și a deplasării. În Figura 3.7 se poate vedea interfața cu numeroasele butoane, iar în Figura 3.8 este mesajul de solicitare a activării conexiunii Bluetooth. Acesta apare atunci când utilizatorul pornește aplicația fără ca să aibă o conexiune Bluetooth pornită. Această interfață mai cuprinde și câteva mesaje de eroare atunci când conexinea nu s-a realizat cu succes sau dispozitivul pereche s-a deconectat.  

Fig. 3.7 Interfața aplicației mobile Fig. 3.8 Mesaj solicitare conexiune Bluetooth

## 3.4 Structuri de baze de date și fișiere

În acest proiect partea de baze de date și fișiere nu are o asa mare relevanță, el putând funcționa destul de bine și fără acestea. Părțile din aplicație care depind de anumite fișiere stocate local(pe post de baze de date) sunt: lucrul în Modul Afișare(*Display Mode*) și distribuirea fișierelor salvate prin email.

Lucrul în *Display Mode*  presupune ca utilizatorul să caute anumite foldere din PC unde se găsesc poze care se vor deschide în interfața principală. Acest mod de lucru este util atunci când nu se dorește neaparat să realizeze capturi prin camera foto, ci doar să se deschidă capturi deja salvate înainte pe PC sau alte capturi/poze. Pozele trebuie să aibă un anumit format(JPEG, PNG etc) specificat de dezvoltatorul aplicației.

Pentru partea de distribuire a pozelor, lucrurile stau în felul următor. Utilizatorul aplicației poate să distribuie unei liste de emailuri pozele salvate temporar în aplicație. Acestea trebuie prima dată salvate într-o anumită locație pe disc. Trimiterea emailului se poate face doar dacă pozele au fost salvate pe disc. La salvarea lor se crează automat o arhivă(care conține aceste poze) într-o locație specificată, iar această arhivă va fi trimisă va fi distribuită listei respective.

## 3.5 Baza materială necesară realizării și testării sistemului

Sistemul din acest proiect este aclătuit din 3 subsisteme, după cum s-a menționat și mai sus(aplicația Desktop cu camera foto, aplicația pe microcontrolerul Arduino și aplicația mobilă). Aplicația pe microcontrolerul Arduino presupune o parte fizică, deci este subsistemul hardware al acestei aplicații. De asemenea o altă parte care poate fi considerată ca și hardware este ansamblul format din camera foto și ecranul secundar care sunt conectate la aplicația desktop.

Pentru acest sistem s-a ales microcontrolerul Arduino Uno, prezentat pe scurt și în capitolul 2, acesta având conectat la pinii lui un modul Bluetooh HC-05. Alte materiale care au fost folosite la acest subsistem au fost rezistoare pentru conectarea modulului Bluetooth și un cablu USB pentru legarea microcontrolerului la PC. De asemenea s-a utilizat și o plăcuță cu sloturi(*breadboard*) pentru o mai ușoară conectare a modului Bluetooth cu Arduino.

Camerele foto folosite sunt legate la PC prin USB și pot să fie camere foto digitale, DSLR(cu funcționalitatea de capturi în timp real), dar și camere web sau alte camere video de la care pot fi preluate cadre. Pentru partea de proiectare, în special pentru acest proiect și pentru cum va fie el folosit, se va folosi un videoproiector conectat la PC. Dar se pot folosi și monitoare sau ecrane secundare, după caz. Testarea sistemului prin aceste echipamente s-a realizat in felul următor. S-au făcut toate conexiunile necesare:

* modulul Bluetooth a fost conectat la microcontrolerul Arduino cu ajutorul unui breadboard, a conectorilor și a unor rezistențe. Acesta este concecatat prin cablul USB la laptop/PC
* camera foto(web/digitală) este conectată la PC împreună cu ecranul secundar
* pentru a vedea dacă Arduino Uno funcționează și primește date de la aplicația mobilă, este folosit monitorul serial(*Serial monitor*) din Arduino IDE.

În Figura 3.9 se pot vedea laptopul, camera web și subsistemul cu Arduino uno ce s-au folosit la aplicație



Fig 3.9 Baza de materiale folosite

# Capitolul 4. Proiectarea Aplicației

În acest capitol se va urmări în principal descrierea implementării acestui sistem(alcătuit din cele 3 aplicații) și metodele de implementare folosite. Se vor descrie, în special pentru aplicația Desktop, diferitele elemente componente ale programului, entități și algoritmi folosiți.

## 4.1 Arhitectura. Descrierea componentelor și a comunicării între ele

Schema-bloc pe care se bazează și arhitectura întregului sistem a fost prezentată și descrisă în capitolul 3, subcapitolul 3.1 în Figura 3.1. Aici se poate vedea întregul sistem și legăturile dintre componente. În acest capitol vor fi analizate si descrise cu un nivel mai jos, componentele subsistemelor. Pentru că aplicația Dekstop este componenta centrală în acest sistem, descrierea se va baza pe această aplicație.



Fig. 4.1 Arhitectura aplicației desktop

În Figura 4.1 este prezentată arhitectura aplicaței desktop. Ea cuprinde anumite componente principale: ferestrele Windows Forms și WPF, în cadrul cărora sunt folosite alte componente sau algoritmi. Legăturile dintre acestea vor fi descrise mai târziu. În cadrul componentei *Fereastra principală* se realizează toate conexiunile dintre componente. Aplicația pornește cu Ecranul de Introducere după care se deschide fereastra principală. În codul din spatele ferestrei principale sunt definite toate evenimentele pe baza căruia rulează această aplicație. După cum s-a menționat și în capitolul 2 această aplicație s-a realizat pe baza paradigmei programării orientate pe evenimente. Astfel, când anumite evenimente se declanșează (click de mouse, tastă apăsată, timp scurs, primire informații prin portul serial etc.) este exectutată o parte de cod numită manipulator de evenimente(*event handlers*).

În aceste evenimente sunt apelate anumite componente și algoritmi definiți în soluția proiectului. În partea stângă a figurii se poate vedea o parte cu anumiți algoritmi. Aceștia sunt apelați din evenimente și fac parte sau nu din aceeași clasă a ferestrei principale. Pentru a fi mai ușor de înteles și pentru a respecta unele principii din programarea orientată pe obiecte, unii algoritmi au fost scriși în clase specializate. Algoritmi precum decuparea pozelor, conversia din formatul Bitmap la Base64 și invers, clonarea pozelor, sunt scriși ca metode statice sau nestatice a unor clase separate.

O altă componentă importantă a aplicației este comunicarea serială care se realizează prin USB cu microcontrolerul Arduino Uno. Acest lucru se realizează în mod asincron, fără ca interfața principala(GUI) să fie afectată sau blocată. Ea se realizează cu ajutorul unor funcții speciale numite *delegate*. Când datele sunt primite de la portul serial, prin delegate se apelează o funcție de *callback.* Datele sunt gândite astfel încât să fie ușor de interpretat de toate cele 3 aplicații, adică cele 3 subsisteme. Aceste comenzi sunt mapate la niște șiruri de caractere care sunt hardcodate din aplicația mobilă. În manipulatoarele de evenimete de la aplicația mobilă se apelează o funcție *SendData(string ).* Aceasta este o funcție generică în care ca și parametru sunt transmise șirurile în care sunt mapate comenzile. Aceste șiruri sunt de forma :

* pentru mărire/micșorare : *zoomin/out[factor]\n*
* pentru deplasare poză : *moveDirecție[factor]\n*
* pentru rotație : *rotateLeft/Right|Degrees\n*

Comenzile se află în prima parte a șirului apoi între paranteze drepte sau între “\ |” se află factorul de multiplicare a deplasării pozelor pe ecran sau gradele de rotație a acestora. Aceste comenzi sunt transmise dispozitivului Bluetooth cu care s-a realizat perechea(HC-05 in acest caz). S-a pus “\n” la sfârșit pentru a trimite fiecare comandă pe rânduri separate. Mai întâi aceste șiruri de caractere sunt transformate în șiruri de bytes și trimise prin funcția *write* a unuiflux de date(*stream*) ce aparține unei prize(*socket*) bluetooth. Un socket este un terminal sau punct final(*endpoint*) ce are rolul de a realiza comunicarea între două entități. Șirul este trimis aplicației de pe microcontroler care îl trimite așa cum l-a primit prin USB aplicației Desktop. În aplicația desktop șirurile acestea sunt interpretate/descompuse pentru a realiza lucrurile “codificate” în ele.

Pentru partea de lucru cu fișiere(pe post de bază de date), o altă componentă din aplicație este cea de importare și exportare a pozelor. Această componentă poate să aibă și o legătură cu partea de distribuire a pozelor prin email, deoarece pozele salvate(exportate pe disc) vor fi cele care vor fi arhivate de către aplicație, și apoi trimise prin email prin protocolul SMTP. Pentru această parte de trimitere a unui email este definită o nouă componentă în soluția aceasta. Este vorba despre o fereastră unde utilizatorul își va introduce datele sale(email și parolă – expeditor) și va trimite arhiva respectivă unei liste de emailuri. Tot utilizatorul crează această listă de emailuri

În partea dreaptă a Figurii 4.1 se poate vedea partea ce include ecranul secundar. Ecranul secundar este pentru program locul unde afișează la început ecranul negru(*blank screen*), iar apoi sunt afișate pozele care pot fi manipulate pe acest ecran. Pentru că pozele se realizează în spatele ferestrei principale(Windwos Forms) și acestea trebuie să fie afișate prin tehnologia mai nouă(WPF – Windows Presentation Foundation), imaginile trebuie să fie convertite. S-a ales modul de lucru cu WPF, pentru că WPF dispune de o tehnologie numită Canvas. Canvas(din spațiul de nume *System.Windows.Controls*) reprezintă o zonă în care se pot poziționa elemente copil într-un control –părinte, folosindu-se coordonatele relative la zona Canvas. Cu Canvas se poate deplasa o fotografie, mări, micșora pe întreg ecranul secundar. După cum s-a spus mai sus, ca o poză să poată fi preluată de WPF- Canvas, ea trebuie convertită.[24]

Pentru această conversie, pozele din Windows Forms trebuies să fie convertite din formatul *Image* la formatul *Base64String*, iar apoi convertite din *Base64String* la formatul *BitmapSource* compatibil pentru WPF. Această conversie este realizată într-o componentă separată din soluția proiectului, iar o instanță din WPF va fi folosită în fereastra principală. Deci se poate vorbi despre o asociere 1 la 1. Acțiunile realizate de WPF sunt: afișarea imaginilor(incluzând ecranul negru) și manipularea acestora prin comenzi care vin de la componenta Windows Forms. Prin manipulare se întelege: mărire, micșorare, rotire în ambele sensuri, deplasare sus, jos, stânga, dreapta a pozelor primite. În această parte de WPF se folosesc anumite clase specifice WPF care au o funcționalitate ideală pentru ceea ce se dorește în acest proiect. În primul rând este vorba despre clasa/controlul ***Canvas****.* Canvas definește o zonă în care se pot plasa elemente copil folosind coordonate care sunt relative la controlul System.Windows.Controls.Canvas.Această clasă Canvas are câteva funcții foarte folositoare pentru a deplasa poza pe ecranul secundar. Aceste funcții sunt : *SetRight, SetLeft, SetTop, SetBottom* care au semnătura de forma : *SetLeft(UIElement element, double length)*. Deci parametrii sunt elementul din interfața grafică pentru care se face deplasarea și lungimea(în pixeli) cu care se va face deplasarea. De asemenea se mai folosește clasa ***Matrix*** care este o matrice de transformare 3x3 folosită pentru transformări în spațiul 2D, împreună cu clasa ***MatrixTransform*** care creează o matrice afină arbitrară folosită pentru a manipula obiecte sau coordonate într-un plan 2D. Acestea din urmă se creează cu ajutorul primei. Transformarea se apilcă unui control din interfața grafică(GUI) atribuind matricea de tipul MatrixTransform proprietății *RenderTransform* a controlului. Lui Matrix i se aplică *ScaleAtPrepend* prin care se realizează mărirea/micșorarea pozelor.[26]



Fig 4.2 Componente aplicație Android

Pentru a putea localiza în orice moment și loc imaginea în controlul Canvas, s-a observat că este nevoie de coordonatele centrului imaginii. Astfel s-a creeat o funcție **Point GetCenter()**  care returnează coordonatele centrului. Cele două coordonate se pot calcula dacă cunosc coordonatele punctelor de la extremitățile : stânga-sus și dreapta-jos și lungimea și înălțimea imaginii. Aceste lucruri pot fi calculate folosind printre altele și funcția *GetContentBounds* a clasei *VisualTreeHelper.*

În figura 4.2 este descrisă arhitectura aplicației mobile și comunicarea cu anumite elemente. Aplicația fiind una simplă, ea are o singură fațadă(*layout*) și o clasă principală *MainActivity.java*. Intrarea în acest subsistem este reprezentată de către comenzile date de către utilizator aplicației care sunt interpretate ca si evenimente. Evenimentele acestea sunt tratate prin funcții speciale, manipulatoare de evenimente. Acestea sunt scrise în clasa MainActivity alături de legăturile dintre GUI și evenimente. Tot aici se realizează și transmiterea comenzilor mai departe, sub formă de șiruri de caractere, prin bluetooth spre Arduino. Deci, în mare parte, și în aplicația mobilă este folosită paradigma programării orientate pe evenimet.

Componenta de program Arduino are doar un rol de intermediar. Ea preia șirul de caractere în care este prezentă comanda de la aplicația mobilă, și îl transmite spre prin USB spre aplicația Desktop.

## 4.2. Cablaje, desene de echipare, alocarea la pinii conectorilor

În Figura 4.3 este prezentată schema electrică a subsistemului alcătuit de modulul Arduino Uno. Aceasta cuprinde și cablajele și alocarea pinilor și rezistențele. Acest subsistem este alcătuit din microcontrolerului Arduino Uno și modulul Bluetooth, prezentate în capitolul 2. Pentru conectarea modulului Bluetooth la Arduino s-au folosit cabluri mamă-tată și mamă-mamă. Cablajul s-a facut astfel:

* pinul RDX de la Bluetooth s-a conectat la pinul 4 de la Arduino(cablul albastru)printr-o rezistență de 1 kΩ. Tot din acest cablu se crează o legătură printr-o rezistență de 2 kΩ la masa(Ground).
* pinul TDX de la Bluetooth s-a conectat la pinul 2 de la Arduino(cablul verde).
* pinul GND –Bluetooth, reprezintă pinul pentru masă și este conectat la pinul GND al Arduino Uno(firul negru).
* pinul VCC – Bluetooth reprezintă pinul pentru alimentarea cu tensiune a modulului Bluetooth. Este legat prin firul roșu la pinul cu alimentare VCC al Arduino Uno.

Modulul Arduino este alimentat prin USB de la PC pentru, și tot prin USB este calea de ieșire din acest subsistem. Prin USB se transmit datele spre PC. Intrarea în subsistem o reprezintă datele primite de la aplicația mobilă, recepționate de modulul Bluetooth HC-05.



Figura 4.3 Schema electrică a modulului Arduino Uno

# Capitolul 5. Realizarea și Utilizarea Sistemului

În acest capitol va fi prezentat modul de utilizare a aplicației. Deoarece aplicația poate să fie folosită în mai multe feluri, se va căuta o generalizare a modului de folosire, lăsând la atitudinea utilizatorului detaliile de utilizare.

**Instalarea Sistemului. Echipamente necesare**

În primul rând, trebuie să fie instalate toate elementele hardware. Pentru aceasta este nevoie de :

- microcontrolerul Arduino Uno cu modulul Bluetooth HC-05. Conexiunile între Ardui-

no și modulul Bluetooth trebuie realizate așa cum s-a prezentat în capitolul 4.2(Figura 4.3). Pentru aceste conexiuni este necesară conectarea cu ajutorul unor conectori tată-mamă și mamă-mamă. De asemenea este nevoie de rezistori de diferite valori, dar care împreună, înseriate să alcătuiască un rezistor de 1 kΩ și altul de 2 kΩ.

- cablu standard USB(Type-B) pentru conectarea plăcuței Arduino Uno la laptop/PC. Cablu HDMI în caz că se folosește un televizor

- un telefon (smart-phone)/tabletă, cu sistemul de operare Android, cu acces la internet pentru instalarea aplicației și cu conexiune Bluetooth

- o cameră foto, video sau web cu posibilitatea de a se realiza capturi în direct cu ea(sau dispozitiv care să fie recunoscut ca un : *Imaging Device*

- un ecran secundar: în funcție de cum se dorește a fi folosit sistemul, acest ecran secundar poate să fie ori un videoproiector, un monitor sau un televizor.

- un PC(laptop sau dekstop PC) pe care rulează Windows, cu 1 port USB disponibil( 2 porturi sunt recomandate în caz că dispozitivul foto se conectează prin cablu USB).

- alte echipamente opționale în funcție de locul în care este folosit sistemul: suport aparat foto, suport videoproiector, prelungitoare cablu USB.

Pentru instalarea aplicației Desktop se va deschide pachetul de instalare și se va instala aplicația. Înainte de deschiderea aplicației se va conecta camera foto/web și ecranul secundar la PC. Dacă ecranul secundar nu este conectat, afișarea se va face implicit pe cel principal. Aplicația depinde foarte mult de mediul în care va fi folosită. Amplasarea camerei, a ecranului(planșă videoproiector sau tabla albă) este esențială pentru ca sistemul să funcționeze cum trebuie.

**Pornirea sistemului. Setările inițiale**

După ce s-au realizat toate conexiunile(aproximativ ca în Figura 3.9), se poate porni aplicația Desktop. Arduino este pornit deja, pentru că este alimentat prin USB și pornește automat, rulând programul încărcat pe el în buclă infinită. Sistemul se poate folosi și în varianta aplicație Desktop + camera foto + ecran secundar, ceea ce implică prezența aproape peramanentă a utilizatorului la PC. În acest caz se vor urma doar instrucțiunile legate de această aplicație, camera foto și de ecranul secundar.

Pentru a putea folosi sistemul, trebuie făcute câteva setări inițiale foarte simple. Aplicația Destkop poate fi folosită în modul captură( *Capture Mode*) sau afișare(*Display mode*). În cele ce urmează se va prezenta lucrul în *Capture Mode.* În mare parte, interfața aplicației Desktop a fost prezentată în capitolul 3 în Figurile 3.3, 3.4, 3.5. Acum se va prezenta și modul de lucru cu acestea. Din Figura 5.1 se poate vedea că pentru a lucra în modul *Capture Mode* trebuie selectat *Combo Boxul* Capture Mode. Dacă o cameră foto compatibilă s-a conectat la PC la *Supported Cameras* ar trebui să apară dispozitivul respectiv. În acest caz:   
“Integrated Camera”. Apoi trebuie selectat modul camerei(*Supported Modes*) după care se selectează butonul *Play* pentru a porni transmisia imaginilor. Apoi în spațiul *Live Camera* va apărea transmisia live de la cameră.

Dacă se dorește ca aplicația să fie utilzată de la distanță, înainte de a da *Play*, se va realiza conectarea aplicației la portul serial(USB) de la care vor veni datele de la aplicația mobilă. Mesajul din josul ferestrei “Phone Connection –Not available” arată că nu s-a realizat conexiunea cu protu serial USB de unde vin datele de la aplicația mobilă. Pentru aceasta se merge la fereastra *Phone Connection*. Aplicația mobilă trebuie instalată pe un dispozitiv smart cu sistemul de operare Arduino. Înainte de a intra în ea trebuie neaparat activat Bluetoothul și realizată conexiunea dispozitivul Bluetooth HC-05. Se caută dispozitivele disponibile în zonă. Pentru aceasta este recomandat ca utilizatorul să nu stea la mai mult de 20 de metri cu telefonul față de Arduino Uno. La conectare, când se asociază dispozitivele, utilizatorul trebuie să introducă **codul Pin 1234**pe telefon/tabletă și apoi să deschidă aplicația.

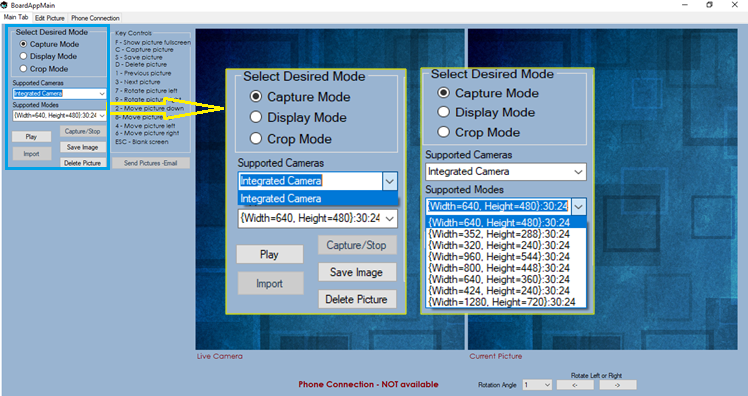
****

Figura 5.1 Setări inițiale, fila principală(*MainTab*) aplicație Desktop

În Figura 5.2 se află fila/fereastradin care se face conexiunea cu aplicația mobilă. De la *Serial Port* se va selecta “Bluetooth Telefon” și se va apăsa connect și dacă conexiunea s-a realizat cu succes, se va afișa un mesajul din dreapta imaginii. Apoi se poate reveni la *MainTab*(Figura 5.1) pentru a începe fotografierea. Se apasă *Play* și acuma aplicația este gata folosire. Pentru a realiza poze cât mai clare, încăperea în care este amplasat sistemul, trebuie să fie nu foarte luminată dar nici foarte întunecoasă. Dacă este prea luminată nu se vor mai distinge bine elementele din poza afișată(dacă e afișată pe tablă), iar dacă este prea întunecosă realizarea pozelor va fi afectată, pozele ieșind întunecoase. Suprafața fotografiată trebuie încadrată aproximativ în centrul imaginii camerei de filmat. Dacă se utilizează un video proiector pentru afișare, imaginea proiectată trebuie să fie încadrată aproximativ în centru vizual al camerei foto.



Figura 5.2 Conectarea la portul serial

**Utilizarea Sistemului**

Comenzi precum *Play, Save, Import, Delete și Capture(fotografiere)* pot fi realizate din interfață direct. Pentru a avea acces la toate funcționalitățile aplicației, este nevoie de folosirea tastelor. Un mic dicționar al tastelor este trecut chiar în interfața principală- *MainTab* pentru a fi ușor de folosit aplicația. Acest dicționar se află în controlul *Key Control.*

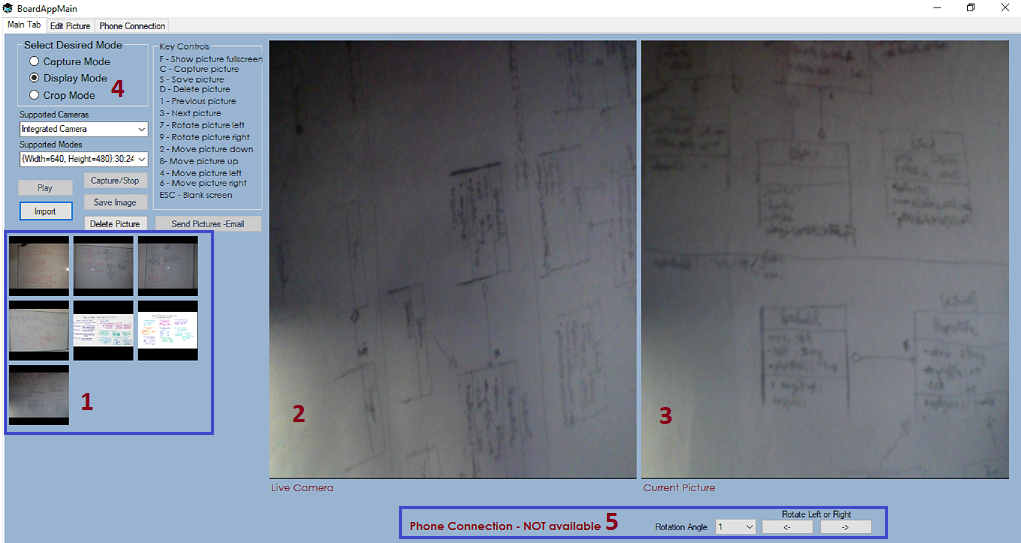
După pornirea camerei foto se apasă butonul C(fotografiere). Imaginea live va fi prezentă în zona notată cu 2 în Figura 5.3. Această poză va apărea în spațiul pentru poza actuală(*Current picture*) în dreapta filei principale. Dacă se dorește păstrarea pozei temporar(adică doar într-o sesiune de rulare a programului) se apasă S: salvare. Pozele salvate vor fi păstrate vizibil în zona 1(un control numit *FlowLayoutPanel* din Figura 5.3). Pentru ștergerea pozei curente se va apăsa tasta D. Poza va dispărea din zona 1. Pentru ca pozele afișate în zona 3 să fie afișate pe ecranul secundar se apasă tasta F (sau NumPad1, NumPad3). Pentru a naviga printre poze se vor folosi tastele Num1-poza anterioară și Num3-poza următoare. Aplicația desktop are mai multe moduri de folosire, modul selectându-se din zona 4. În modul captură - Capture Mode se lucrează direct cu camera foto, pozele făcute fiind și cele salvate și afișate. După selecția acestuia din zona 4, butoanele *Play* și *Capture/Stop* vor fi activate. În modul afișare *Dispaly Mode* pozele pot să fie importate de la anumite locații. După selecția acestuia din zona 4, butonul Import va fi activat. La apăsarea acestuia, se va deschide o altă interfață(Figura 5.4). În această interfață utilizatorul trebuie să selecteze un folder unde programul va căuta toate pozele cu o anumită extensie și le va deschide în controlul din stânga.****

Figura 5.3 Interfața principală-Aplicația Desktop

Când aplicația este pornită, pe ecranul secundar este afișată o imagine neagră. La apasarea butonului F aici apare imaginea din zona 3. La apăsarea tastei ESC, se revine oricând la ecranul negru. Pentru manipularea pozelor pe ecranul secundar se vor descrie în continuare comenzile. Mărirea și micșorarea pozelor se realizează de la butoanele + și - .Rotirea pozei în sensul acelor de ceasornic se face prin NumPad9 și invers prin NumPad7. Tranziția pozelor în cele 4 direcții este astfel: NumPad2 – jos, NumPad8- sus,NumPad4 – stânga,NumPad6 – dreapta.

Pentru funcționalitatea de distribuire a pozelor făcute unei anumite liste de emailuri, s-a realizat interfața din Figura 5.5. Aceasta se deschide la apăsarea butonului “*SendPictures- Email”* din interfața principală. Pentru a folosi această funcționalitate este necesară conexiunea la internet. Înainte de deschiderea interfeței trebuie apăsat butonul *Save Images* pentru a salva pozele din zona 1 într-o arhivă pe disc. Apoi se selectează butonul *Send Pictures – Email* moment în care este deschisă interfața. Aici se adaugă emailuri prin butonul *Add Email*  de la spațiul *Type Email.* De la butonul *Remove Email* se poate sterge emailul selectat din listă. Expeditorul emailului se trece în partea dreaptă(email și parolă) după care se trimite emailul prin apăsarea *Send Email.*

Fig. 5.4 Interfață importare poze Fig. 5.5 Interfață distribuire poze

În zona 5 (Figura 5.3) în *MainTab* se află un mesaj: ”Phone Connection – NOT available” prin care se poate vedea starea conexiunii cu portul serial – USB. Prin acesta se primesc date pe canalul : dispozitiv smart – Bluetooth – Arduino – USB – PC . După ce se realizează conectarea la portul USB, mesajul afișat o să fie: “Phone CONNECTED”. Tot în această zonă există un control Combo Box de unde se poate selecta numărul de grade(0 – 360 ) cu care va fi rotită poza **doar** pe ecranul secundar la tastarea NumPad7 și NumPad9. Pentru că pozele importate pot avea diferite formate și metadate diferite, există cazuri în care acestea se deschid în zona 3 rotite în diferite părți, la 90, 180 de grade. Astfel, există 2 butoane pentru a roti în cele două sensuri la 90 grade poza din zona 3, butoane situate sub labelul “Rotate left or right”.

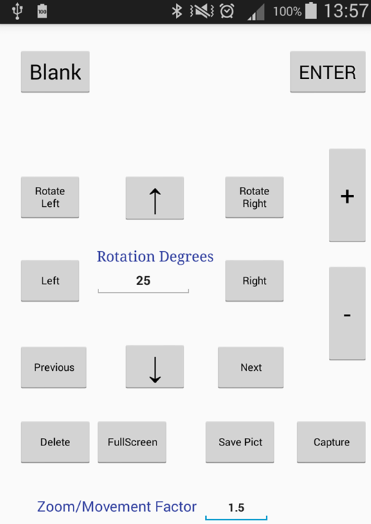
 

Fig. 5.6 Interfață distribuire poze Fig. 5.7 Aplicație mobilă

Aplicația desktop are 3 file(*tabs)*: *MainTab, Edit Picture, Phone Connection.* Prima filă a fost prezentată mai sus. A doua filă( Figura 5.6) are funcționalitatea de a permite utilizatorului să modifice luminozitatea și contrastul pozei. Poza din această filă este poza care în momentul clickului pe filă era afișată în zona 3 din Figura 5.3. După editare se apasă *Save Changes* și se merge înapoi la fila principală unde în zona 3 apare fotografia editată. Pentru a păstra aceste modificări, se recomandă salvarea pozei cel putin temporar(tasta S sau butonul *Save* din interfață).

În Figura 5.7 se poate vedea interfața simplă a aplicației mobile cu butoanele și spațiul text pentru introducerea gradelor cu care se dorește a fi rotită imaginea. De la butonul *Blank* se afișează pe ecranul secundar imaginea neagră. Prin butonul *Enter* utilizatorul poate să închidă mesajele de atenționare/eroare care vor apărea pe ecranul secundar și care provin de la aplicația Desktop. De exemplu dacă utilizatorul încearcă să facă o captură, iar camera foto nu este pornită, pe ecranul secundar va apărea mesajul din Figura 5.8. Apoi, prin apăsarea butonului *Enter*, acest mesaj va putea fi închis de la distanță. De asemenea, mai există și alte mesaje de tip dialog în aplicație care pot fi accesate doar direct de la PC. Unul din ele este în Figura 5.9.

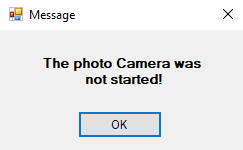
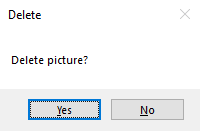
 

Fig. 5.8 Mesaj de atenționare Fig. 5.9 Fereastră de tip Dialog -Ștergere

# Capitolul 6. Concluzii

Proiectul prezentat în această lucrare are rolul de a contribui în automatizarea și schimbarea unor vechi metode de predare cu unele mai eficiente. El își propune să nu implice o schimbare radicală a metodelor, dar într-o măsură temperată să fie de bun augur atât pentru cei care o folosesc, dar și pentru cei care se bucură de utilizarea ei de către alte persoane. El reprezintă un început al aplicării unei idei, care în funcție de cum va continua dezvoltarea proiectului, va avea mai mult sau mai puțin succes. În ceea ce privește modul lui de dezvoltare, a fost creat în totalitate de un singur programator, într-o perioadă destul de scurtă de timp pentru o astfel de aplicație. Din această cauză proiectul nu poate avea o funcționalitate foarte complexă și o funcționare perfectă. Pentru ca aceste lucruri să fie realizabile, este nevoie de mult mai mult timp pentru dezvoltarea și testarea aplicației alături de o echipă.

În această lucrare, varianta sistemului prezentată a fost mai mult cea cu videoproiector și tablă albă, aceasta fiind cea mai probabilă formă a ei de utilizare, ținând cont de ce scrie în [1]. Pentru această formă de utilizare și nu numai, s-a constatat în urma testării sistemului cu diferite camere că un lucru esențial în fotografierea succesivă a tablei este calitatea camerei foto. Când se spune cameră foto se referă la orice dispozitiv care transmite imagini(chiar și live) spre PC, deci și camerele video și web. O rezoluție cât mai mare ajută la distingerea mai ușoară a detaliilor(în special fiind vorba aici de scheme arhitecturale). Dar o cameră foto cât mai bună(calitate poză + rezoluție mare) este implicit mai scumpă. Unul din scopurile acestui proiect este să aibă costuri minime pentru a putea fi folosit în detrimentrul tablelor digitale(*smart-boardurilor*). Camerele foto pentru care această versiune aplicației a fost construită sunt cele care sunt recunoscute de către Windows ca și *Imaging Devices*. Aplicația s-a testat cu camere web. Camerele web fiind concepute pentru filmat și nu pentru fotografiat, nu se poate obține o calitate foarte bună a pozelor, dar costurile ei de achiziție sunt mici. Cealalată variantă ar fi fost folosirea camerelor foto care suportă funcționalitatea de fotografiere live(sau *Tetherd Shooting*). Dar dispozitivele care au această funcționalitate sunt în general cele mai recente variante de camere DSLR(Nikon,Cannon) care au costuri destul de semnificative și folosirea lor nu este rentabilă.

De asemenea, sa observat că mediul în care este folosită aplicația și materialele folosite influențează în mod fundamental rezultatele folosirii aplicației. Luminozitatea încăperii, sursa și direcția luminii naturale sau artificiale, reflexiile, suprafața pe care se proiectează, toate contribuie la un rezultat bun sau rău al folosirii sistemului. Blițul camerei foto este de asemenea un factor care influențiază calitatea pozelor. Folosit corect, poate aduce un plus sistemului. Pentru rezultate cât mai bune s-a constatat că așezarea camerei foto este importantă. S-a observat că tabla fotografiată trebuie să fie aproximativ în central imaginii, iar camera să fie perpendiculară și pe centrul tablei. Din cauza acestor lucruri trebuie să se facă anumite compromisuri care nu pot fi delmitate clar din cauza multitudinii de situații particulare în care sistemul poate să fie folosit.

**Perspectiva de viitor**

Se poate spune că aplicația aceasta are o perspectivă bună de viitor, pentru că s-au găsit multe posibilități de îmbunătățire și de extindere a funcționalităților ei. Acest lucru se bazează și pe faptul că industria IT este un domeniu foarte dinamic în care tot mai frecvent apar tehnologii noi. Unele tehnologii care acum nu sunt așa accesibilie(precum realizarea pozelor live - *Tetherd Shooting*) în viitor vor fi ceva uzual și probabil vor fi implementate. Se dorește ca în viitor să se implementeze în aplicația desktop algoritmi de procesare a imaginilor mai performanți. Prin aceștia se vor putea extrage dintr-o poză doar anumite puncte de interes, precum tabla albă. Pentru că amplasarea camerei nu va fi ușoară în orice încăpere s-a gândit o soluție pentru a avea control asupra camerei și după amplasare. Ea va fi montată pe niște motorașe servo care vor funcționa ca o articulație. Acestea pot fi legate la Arduino și controlate prin aplicația mobilă și desktop pentru a roti camera în orice direcție. De asemenea se poate implementa o funcționalitate ca utilizatorul să poată să deseneze peste pozele făcute. Deci, după cum se vede și mai sus, aplicația această nefiind o aplicație pur software, are multe variabile de mediu și de echipament folosit de care depinde. Folosirea sistemului cu success depinde de aceste lucruri prezentate.

Prin aces proiect, s-a dorit rezolvarea unei situații sau probleme aparute în viata reală, folosind programarea și alte cunoștințe specifice de IT. Acestea au tocmai acest rol: de a îmbunătăți viața și activitatea omului. Acest lucru a fost reușit într-o măsură relativ bună, ținând cont de circumstanțe, dezoltarea lui continuând și în prezent.

# Capitolul 7. Bibliografie

[1] Phd. Lect. Arch. Camil O. Milincu, Phd. Arch. Otilia A. Tudoran, *Whiteboard upgrade? Discussing specific needs for architecture and design*, Politehnica University Timișoara – Faculty of Achitecture, Romania

[2] Dorin Berian, *Medii și tehnologii de programare*, capitolul 1*: Introducere în Visual Studio .Net*

[3] <https://en.wikipedia.org/wiki/.NET_Framework>

[4] Andrew Troelsen, Philip Japikse, *C# 6.0 and the .NET 4.6 Framework, Seventh Edition*, Apress, New York, 2015

[5] <https://en.wikipedia.org/wiki/Microsoft_Visual_Studio>

[6] John Paul Mueller, Bill Sempf, Chuck Sphar, *C# 7.0 Book All-in-one For Dummies*, John Wiley & Sons Inc, Hoboken, New Jersey Media, 2018

[7] Andrew Troelsen, Philip Japikse, *Pro C# 7 – With .NET and .NET Core - Eighth Edition,* Apress, New York, 2017

[8] S.Hansen, Timothy V. Fossum *Event Based Programming.* Kenosha WI, 2010

[9] <https://en.wikipedia.org/wiki/Event-driven_programming>

[10] <http://www.aforgenet.com/framework/>

[11] <http://www.emgu.com/wiki/index.php/Main_Page>

[12] <https://www.farnell.com/datasheets/1682209.pdf>

[13] Bluetooth wireless technology basics –HP - <http://www.hp.com/ctg/Manual/c00186949.pdf>

[14] <https://www.gme.cz/data/attachments/dsh.772-148.1.pdf>

[15] <https://components101.com/wireless/hc-05-bluetooth-module>

[16] Chryssa Aliferi, *Android Programming Cookbook*, Exelixis Media P.C , 2016

**[**17] Android ATC Team , *Java Fundamentals for Android­­ Developmenet,*Android ATC, 2017

[18] Richard L. Halterman *Fundamentals of C++ Programming,* Southern Adventist University, 2018

<http://python.cs.southern.edu/cppbook/progcpp.pdf> - Carte C++

[20] Jan Axelson, *USB Complete* – Jan Axelson, 2005

[21] Jay Trodden, Donovan Anderson *USB 3.0 Technology*, MindShare Inc., 2013

[22] <https://docs.microsoft.com/en-us/previous-versions/msp-n-p/jj159335(v=pandp.10)>

[23] <http://condor.depaul.edu/sjost/hci430/documents/testing/UnitTesting.pdf>

[24] <https://msdn.microsoft.com/en-us/library/system.windows.controls.canvas(v=vs.110).aspx>

[25] Andrew S. Tanenbaum, David J. Wetherall, *Computer Networks*, Pearson, 2011

[26] <https://msdn.microsoft.com>

**Imagini :**

**[img1]** Dorin Berian, *Medii și tehnologii de programare*

**[img2]** <http://www.wilsonmar.com/msdotnet.htm>

**[img3]** Dorin Berian, *Medii și tehnologii de programare*

**[img4]** <https://www.slideshare.net/ashwiniawatare/event-oriented-programming>

**[img5]** S.Hansen, Timothy V. Fossum *Event Based Programming.* Kenosha WI, 2010

**[img6]** [https://www.google.ro/search?biw=1280&bih=893&tbm=isch&sa=1&ei=a9L7Wtu\_H5S jmwXP2ILABQ&q=opencv+architecture&oq=OpenCV+architec&gs\_l=img.3.0.0i19k1l2.9555.11066.0.11901.9.6.0.3.3.0.140.383.0j3.3.0....0...1c.1.64.img..3.6.404....0.4-hQfthNgbg#imgrc=mbENGINorYG7QM](https://www.google.ro/search?biw=1280&bih=893&tbm=isch&sa=1&ei=a9L7Wtu_H5S%20jmwXP2ILABQ&q=opencv+architecture&oq=OpenCV+architec&gs_l=img.3.0.0i19k1l2.9555.11066.0.11901.9.6.0.3.3.0.140.383.0j3.3.0....0...1c.1.64.img..3.6.404....0.4-hQfthNgbg#imgrc=mbENGINorYG7QM)

**[img7]** [https://www.google.ro/search?q=arduino+uno&source=lnms&tbm=isch&sa=X&ved=2 ahUKEwjf8\_2MzorbAhXLtxQKHR4jDgwQ\_AUoAXoECAEQAw&biw=1280&bih=893#imgrc=5fhM55F\_QIl56M](https://www.google.ro/search?q=arduino+uno&source=lnms&tbm=isch&sa=X&ved=2%20ahUKEwjf8_2MzorbAhXLtxQKHR4jDgwQ_AUoAXoECAEQAw&biw=1280&bih=893#imgrc=5fhM55F_QIl56M):

<https://www.google.ro/search?q=arduino+uno&source=lnms&tbm=isch&sa=X&ved=2ahUKEwjf8_2MzorbAhXLtxQKHR4jDgwQ_AUoAXoECAEQAw&biw=1280&bih=893#imgrc=tyWNFG6xZWQw5M>:

**[img8]** <https://www.google.ro/search?q=bluetooth+hc05&source=lnms&tbm=isch&sa=X&ved=2ahUKEwii04OMz43bAhVHjqQKHbOrBxwQ_AUoAXoECAEQAw&biw=1280&bih=844#imgrc=XvoZOa-UoXBIJM>:

[**img9**] Android ATC Team, *Java Fundamentals for Android­­ Developmenet,*Android ATC,2017



1. Prima pagină trebuie făcută exact cum scrie in PRECIZĂRI PRIVIND ELABORAREA ŞI PREZENTAREA LUCRĂRII DE LICENȚĂ /PROIECTULUI DE DIPLOMĂ ?
2. Plagiat

AIAlogo.bmp