**<Lescai Valentin-Mihai>**

# Documentul de proiectare

**Cuprins**

[**1.**](#_heading=h.4tkfbxda5smc) **Introducere 1**

[1.1](#_heading=h.ddxhrvegl58z) Scopul documentului 1

[**2.**](#_heading=h.z9mlpbc9imlw) **Prezentare generală și abordări de proiectare 2**

[2.1](#_heading=h.8rg1t8d16u2t) Prezentare generală 2

[2.2](#_heading=h.s57fd910r97g) Presupuneri/ Constrângeri/ Riscuri 2

[2.2.1](#_heading=h.laz9s6fu9ozy) Presupuneri 2

[2.2.2](#_heading=h.ij358js8zmkb) Constrângeri 2

[2.2.3](#_heading=h.igdwjftzk7et) Riscuri 3

[**3.**](#_heading=h.egb6upvbd21l) **Considerațiii de proiectare 4**

[3.1](#_heading=h.jn8ksidn8zze) Obiective și linii directoare (ghiduri) 4

[3.2](#_heading=h.hixmu7o3lmhi) Metode de dezvoltare 4

[3.3](#_heading=h.qbjakdgz5102) Strategii de arhitectură 4

[**4.**](#_heading=h.452g08zhfvub) **Arhitectura Sistemului și Proiectarea Arhitecturii 6**

[4.1](#_heading=h.qpcfugaq9zdb) Vedere logică 6

[4.2](#_heading=h.huqs3if9ghg) Arhitectură hardware 6

[4.3](#_heading=h.e8dkin5lsdy) Arhitectură software 6

[4.4](#_heading=h.5g8ooybupenp) Arhitectura informațiilor 7

[4.5](#_heading=h.61pnaolra4gc) Arhitectura de comunicații interne 7

[4.6](#_heading=h.hdat85rdkrqc) Diagrama de arhitectură a sistemului 8

[**5.**](#_heading=h.9pg2327ljpuw) **Proiectarea sistemului 9**

[5.1](#_heading=h.h710flmo0v7b) Proiectarea bazei de date 9

[5.1.1](#_heading=h.nndtiv2rxsxy) Obiecte de date și structuri de date rezultante 9

[5.1.2](#_heading=h.ggx19s823kfo) Fișiere și baze de date 9

[5.2](#_heading=h.ri6df44jdtt6) Conversii de date 9

[5.3](#_heading=h.e8m8ja6efmqh) Interfețe utilizator 10

[5.3.1](#_heading=h.mhp5s9m8kqtp) Intrări 10

[5.3.2](#_heading=h.kgu0s8v8k2o9) Ieșiri 10

[5.4](#_heading=h.k93izgjrttlk) Proiectarea interfețelor cu utilizatorul 10

[**6.**](#_heading=h.92lvozkwnam) **Scenarii de utilizare 11**

[**7.**](#_heading=h.rihrq0lp6am5) **Proiectare de detaliu 12**

[7.1](#_heading=h.4ax23ye0h7xu) Proiectare hardware de detaliu 12

[7.2](#_heading=h.k4fgf17i1nnx) Proiectare software de deatliu 12

[7.3](#_heading=h.jrhs4q51vixi) Proiectare detaliată de securitate 13

[7.4](#_heading=h.26alq0q9brhi) Proiectare de detaliu pentru performanța sistemului 14

[7.5](#_heading=h.icpxyct7rax2) Proiectare detaliată a comunicațiilor interne (între componente) 14

[**8.**](#_heading=h.knbzar13ebu0) **Controale pentru verificarea integrității sistemului 15**

[**Anexa A: Gestiunea modificărilor documentului 16**](#_heading=h.lkh08jgmi0pi)

[**Anexa B: Acronime 17**](#_heading=h.d4aqabv4aoh0)

[**Anexa C Documente la care se face referire 18**](#_heading=h.5xridibdm7sk)

## Introducere

### Scopul documentului

Documentul de proiectare descrie arhitectura și design-ul detaliat al sistemului pentru analiza și detectarea statisticilor în tenis, care include funcționalitățile de detectare a jucătorilor și mingii, calculul vitezei mingii și distanței parcurse de jucători, și vizualizarea acestora pe un teren grafic.

## Prezentare generală și abordări de proiectare

### Prezentare generală

*Sistemul va fi un software de procesare a videoclipurilor care utilizează algoritmi de viziune computerizată și învățare automată pentru a detecta jucătorii și mingea, pentru a calcula statistici despre viteza mingii și distanța parcursă de jucători, și pentru a vizualiza aceste informații pe un teren grafic. Sistemul va fi compus din mai multe module independente care comunică între ele pentru a oferi o experiență interactivă și rapidă.*

### Presupuneri/ Constrângeri/ Riscuri

#### Presupuneri

*Se presupune că videoclipurile vor fi capturate în condiții favorabile, cu o rezoluție minimă de 720p.*

*Se presupune că utilizatorii vor avea acces la hardware-ul necesar (GPU dedicat).*

#### Constrângeri

* *Sistemul trebuie să ruleze pe platforme Windows sau Linux și să fie compatibil cu cele mai recente versiunile de Python și biblioteci AI (TensorFlow, OpenCV).*
* *Video-ul introdus este de aproximativ 10 secunde, iar rezoluția minimă acceptată va fi 720p.*

#### Riscuri

* *Posibile erori de procesare datorită calității slabe a videoclipurilor (mișcare rapidă, iluminate slabe, etc.).*
* *Riscuri legate de performanța hardware-ului în cazul în care nu există un GPU dedicat suficient de performant.*

## Considerații de proiectare

* **Procesarea video-urilor în timp real:** Sistemul trebuie să poată analiza și procesa videoclipuri de 10 secunde la o frecvență de cadre de cel puțin 30 fps, fără a întâmpina întârzieri semnificative. Acest lucru poate pune presiune pe resursele hardware, mai ales când se utilizează tehnici de învățare automată pentru detectarea jucătorilor și mingii.
* **Scalarea terenului grafic:** Conversia coordonatelor din pixeli în unități de măsură reale (metri) pentru terenul grafic necesită o scalare corectă, iar eroarea în acest proces poate afecta acuratețea vizualizării.
* **Acuratețea detecției AI:** Modelele YOLOv8 și YOLOv5 sunt deja antrenate, dar performanța lor va depinde mult de calitatea videoclipului introdus. Se vor face ajustări fine ale modelului, dacă este necesar, pentru a îmbunătăți detectarea în condiții de iluminare diferite și cu mișcări rapide.
* **Eficiența calculării statisticilor:** Calculul vitezei mingii și al distanței parcurse de jucători trebuie să fie rapid și precis, fără a compromite performanța vizualizării terenului grafic.

### Obiective și linii directoare (ghiduri)

Obiectivele principale ale designului sunt de a asigura performanță, precizie, o interfață intuitivă, scalabilitate și compatibilitate hardware. Sistemul trebuie să proceseze videoclipuri în timp real sau aproape în timp real, fără întârzieri semnificative în detectarea jucătorilor și mingii sau în calcularea statisticilor. Aceste date trebuie să fie exacte și consistente, iar interfața utilizatorului trebuie să fie ușor de utilizat și clară, permitând vizualizarea rapidă a mișcărilor jucătorilor și mingii, precum și a statisticilor relevante.

Designul va respecta câteva linii directoare importante. În primul rând, codul va fi modular, fiecare componentă fiind dezvoltată și testată separat pentru a permite întreținerea ușoară și îmbunătățirea sistemului pe viitor. Reutilizarea componentelor software existente, cum ar fi modelele YOLOv5 și YOLOv8, va ajuta la economisirea timpului de dezvoltare și va asigura performanță excelentă în detectarea obiectelor.

În ceea ce privește dezvoltarea interfeței, aceasta va respecta ghiduri de accesibilitate pentru a asigura că utilizatorii cu deficiențe de vedere sau alte dificultăți nu vor întâmpina probleme în utilizarea sistemului. În plus, vom opta pentru un design curat și intuitiv, care să fie accesibil chiar și pentru utilizatorii fără experiență tehnică.

De asemenea, vom pune un accent deosebit pe eficiența resurselor, având în vedere că procesarea video-urilor poate fi intensivă. De aceea, sistemul va fi optimizat pentru a rula eficient pe hardware-ul recomandat, în special pe GPU dedicat.

### Metode de dezvoltare

În ceea ce privește metodele formale, vom folosi UML (Unified Modeling Language) pentru a crea diagrame de secvență, diagrame de clase și diagrame de activități care vor descrie interacțiunile dintre modulele de procesare video, calculul statistic și vizualizarea grafică.

Abordarea alternativă luată în considerare a fost utilizarea unei arhitecturi monolitice, însă aceasta a fost respinsă din cauza complexității suplimentare pe care o impune integrarea strânsă a tuturor componentelor. În plus, o arhitectură monolitică ar fi avut un impact negativ asupra performanței, deoarece toate componentele ar fi fost procesate pe același thread, fără posibilitatea de a le paraleliza.

Dacă se vor întâmpina dificultăți în procesarea video-urilor de calitate scăzută, soluțiile de rezervă includ aplicarea unor tehnici de filtrare a imaginii pentru îmbunătățirea calității și adaptarea modelelelor AI pentru a le face mai robuste în fața variabilităților de mediu.

### Strategii de arhitectură

În ceea ce privește arhitectura sistemului, am ales o abordare modulară care permite separarea clară a fiecărei funcționalități, precum procesarea video, calculul statistic și vizualizarea grafică. Această abordare va facilita întreținerea sistemului pe termen lung și va permite dezvoltarea de noi funcționalități, cum ar fi analiza mișcării echipei sau integrarea cu alte sisteme de analiză video.

Raționamentul din spatele acestei alegeri este acela de a oferi flexibilitate și scalabilitate, având în vedere că este posibil ca în viitor să dorim să extindem sistemul pentru a analiza jocuri de echipă sau pentru a adăuga mai multe statistici. Această arhitectură modulară va permite extinderea ușoară a funcționalităților fără a afecta structura generală a sistemului.

În ceea ce privește deciziile tehnologice, am ales să folosim YOLOv5 și YOLOv8 pentru detecția jucătorilor și mingii, datorită acurateței și performanței lor excelente în detecția obiectelor, și vom continua să folosim aceste modele fără a face modificări majore, pentru a economisi timp și resurse. Dacă se va considera necesar, vom face ajustări fine ale modelului pentru a-l adapta mai bine la specificul tenisului.

## Arhitectura Sistemului și Proiectarea Arhitecturii

### Vedere logică

Vederea logică a arhitecturii reflectă modul în care modulele sistemului sunt organizate pentru a îndeplini funcționalitățile solicitate. Componentele principale ale sistemului sunt:

1. Modulul de procesare video:

* Acesta este primul modul care se va ocupa cu încărcarea fișierului video și procesarea acestuia cadru cu cadru.
* Va aplica modelele de detecție YOLOv8 (pentru detectarea jucătorilor) și YOLOv5 (pentru detectarea mingii), utilizând OpenCV pentru manipularea imaginii și analiza video-ului.
* Datele extrase, cum ar fi coordonatele jucătorilor și ale mingii din fiecare cadru, vor fi trimise direct la modulul de calcul al statisticilor.

1. Modulul de calcul al statisticilor:

* Acesta va prelua datele de poziție ale jucătorilor și mingii și va calcula statisticile necesare, cum ar fi viteza mingii și distanța parcursă de jucători.
* Calculul vitezei mingii va fi realizat pe baza diferențelor de poziție între cadre consecutive, iar distanța parcursă de jucători va fi calculată pe baza mișcării lor între cadrele video.

1. Modulul de vizualizare grafică:

* Acesta va desena terenul de tenis și va actualiza continuu pozițiile jucătorilor și mingii, pe măsură ce se procesează fiecare cadru al videoclipului.
* Va crea și actualiza terenul grafic, oferind o interfață vizuală în care mișcările jucătorilor și ale mingii sunt redată în timp real.
* Pe terenul grafic vor fi afișate și statisticile relevante, cum ar fi viteza mingii și distanța parcursă de jucători.

Comunicarea între module se va face direct, prin apeluri de funcții și manipularea variabilelor interne, iar datele vor fi transmise rapid între componente. Astfel, vom obține un flux de lucru eficient și o performanță optimizată.

### Arhitectură hardware

Arhitectura hardware a sistemului va fi optimizată pentru a asigura procesarea video-urilor și rularea modelelor de învățare automată la o viteză acceptabilă. Hardware-ul necesar pentru sistem este:

* CPU: Un procesor de cel puțin Intel i5 sau echivalent AMD Ryzen 5, care să asigure procesarea rapidă a datelor.
* GPU: O placă video dedicată de tip NVIDIA GTX 1060 sau mai bună, care va accelera procesarea video-urilor și va asigura rularea eficientă a modelelor YOLO.
* Memorie RAM: Min. 8 GB RAM pentru a susține execuția rapidă a algoritmilor de procesare a imaginilor și a datelor.
* Spațiu de stocare: Un spațiu de 50 GB, preferabil pe un SSD, pentru a stoca fișierele video și datele procesate într-un mod rapid și accesibil.
* Monitor: Un monitor cu rezoluție minimă Full HD (1920x1080) pentru vizualizarea clară a terenului grafic și a statisticilor.
* Sistemul va fi capabil să ruleze pe o platformă de tip desktop sau laptop, dar va beneficia cel mai mult de pe urma unei configurații cu GPU dedicat.

### Arhitectură software

Arhitectura software va fi organizată pe module, fiecare având o responsabilitate bine definită, iar comunicarea între module va fi realizată intern, fără a depinde de API-uri externe. Modulul de procesare video va fi responsabil pentru prelucrarea fișierelor video și aplicarea algoritmilor de detectare YOLO, folosind OpenCV și TensorFlow pentru procesarea imaginilor și a video-urilor.

Modulul de calcul al statisticilor va prelua datele despre jucători și minge și va calcula viteza acestora și distanța parcursă, folosind algoritmi matematici pentru a determina aceste statistici pe baza mișcărilor detectate în videoclip.

În final, modulul de vizualizare grafică va actualiza terenul de tenis pe măsură ce se procesează fiecare cadru video, afișând jucătorii și mingea în mișcare, precum și statisticile relevante.

Componentele software vor comunica prin funcții interne și variabile partajate, iar procesarea va fi realizată în paralel pe fiecare cadru video pentru a asigura performanță în timp real.

### Arhitectura informațiilor

Informațiile procesate de sistem vor include coordonatele jucătorilor și mingii, viteza mingii, distanța parcursă de jucători și alte date relevante despre meciuri. Aceste informații vor fi stocate în variabile interne, fără a utiliza baze de date externe. Coordonatele și statisticile vor fi transmise între module pentru a fi procesate și vizualizate în timp real.

De asemenea, istoricul meciurilor și statisticilor vor fi stocate temporar în memorie pentru a fi accesibile pe durata sesiunii de utilizare a aplicației. Aceasta va permite utilizatorului să vizualizeze datele istorice fără a depinde de o bază de date externă.

### Arhitectura de comunicații interne

Comunicarea între modulele software va fi realizată fără API-uri externe, prin transferul direct al datelor între componente folosind funcții și variabile partajate. Fluxul de date va fi gestionat astfel:

Modulul de procesare video va trimite coordonatele detectate ale jucătorilor și mingii către modulul de calcul al statisticilor.

Modulul de calcul al statisticilor va trimite valorile pentru viteza mingii și distanța parcursă către modulul de vizualizare grafică.

Modulul de vizualizare grafică va actualiza terenul de tenis și va afișa statisticile pe ecran, asigurând o actualizare rapidă și fără întârzieri semnificative.

## Proiectarea sistemului

### Conversii de date

În acest sistem, conversiile de date joacă un rol important în gestionarea și transformarea datelor de la formatul inițial în care sunt procesate (de exemplu, coordonatele din pixeli ale jucătorilor și mingii în fiecare cadru video) într-un format care poate fi utilizat pentru a calcula statistici și pentru a vizualiza mișcările pe terenul grafic.

**Conversia coordonatelor din pixeli în unități reale (metri)**

Una dintre cele mai importante conversii de date este transformarea coordonatelor detectate din pixeli (care sunt unități de măsură pentru imagine) în unități de măsură reale (de exemplu, metri) care reprezintă distanțele pe terenul de tenis. Această conversie este necesară pentru a putea calcula statistici relevante, cum ar fi viteza mingii sau distanța parcursă de jucători.

### Interfețe utilizator

Interfața cu utilizatorul (UI) este componenta care le permite utilizatorilor să interacționeze cu sistemul. În cazul acestui proiect, interfața trebuie să fie clară, intuitivă și ușor de utilizat, având ca scop vizualizarea datelor în timp real, precum mișcările jucătorilor și ale mingii pe terenul grafic și afișarea statisticilor (viteza mingii, distanța parcursă de jucători etc.).

UI-ul va include mai multe elemente vizuale esențiale:

* Terenul grafic: O reprezentare a terenului de tenis pe care vor fi desenate mișcările jucătorilor și ale mingii, în funcție de coordonatele calculate. Mișcările vor fi actualizate în timp real pe măsură ce se procesează fiecare cadru din videoclip.
* Statistici vizualizate: Sub terenul grafic, vor fi afișate statistici precum:
  + Viteza mingii în kilometri pe oră (km/h).
  + Distanța parcursă de jucători în metri.

## Scenarii de utilizare

Scenariile de utilizare descriu pașii specifici pe care utilizatorii îi vor parcurge pentru a interacționa cu sistemul și pentru a obține rezultatele dorite. Aceste scenarii ajută la înțelegerea comportamentului sistemului în diferite situații, precum încărcarea videoclipurilor, procesarea acestora și vizualizarea statisticilor.

**Scenariul 1: Încărcarea și procesarea unui videoclip**

Actor principal: Utilizatorul (antrenor, analist, jucător)

Descriere: Utilizatorul încarcă un videoclip de 10 secunde cu un meci de tenis, iar sistemul va procesa acest videoclip pentru a detecta jucătorii și mingea, calculând viteza mingii și distanța parcursă de jucători.

Pași:

1. Utilizatorul deschide aplicația și accesează ecranul principal.
2. Utilizatorul apasă butonul „Încarcă Video”.
3. Apare un dialog pentru selectarea fișierului video de pe dispozitivul utilizatorului.
4. Utilizatorul selectează fișierul video și apasă „Deschide”.
5. Sistemul începe să proceseze videoclipul cadru cu cadru.
6. În timpul procesării, utilizatorul poate vedea un mesaj de progres sau o bară de încărcare.
7. După finalizarea procesării, sistemul detectează jucătorii și mingea și calculează viteza mingii și distanța parcursă de jucători.
8. Rezultatele sunt afișate pe terenul grafic, iar statisticile (viteza mingii și distanța parcursă) sunt prezentate sub terenul grafic.

**Scenariul 2: Încărcarea și procesarea unui videoclip**

Actor principal: Utilizatorul (antrenor, analist, jucător)

Descriere: După procesarea videoclipului, utilizatorul dorește să vizualizeze mișcările jucătorilor și mingii pe terenul grafic și să analizeze statisticile (viteza mingii și distanța parcursă de jucători).

Pași:

* 1. După procesarea videoclipului, sistemul desenează terenul de tenis și plasează jucătorii și mingea pe teren pe baza coordonatelor detectate.
  2. Utilizatorul poate vizualiza mișcarea jucătorilor și mingii pe terenul grafic.
  3. Statisticile relevante (viteza mingii și distanța parcursă) sunt afișate sub terenul grafic.
  4. Terenul grafic și statisticile vor fi actualizate în timp real pe măsură ce videoclipul este redat.

## Proiectare de detaliu

### Proiectare hardware de detaliu

**Procesor (CPU)**:

* Sistemul va necesita un procesor performant, de preferință **Intel i5** sau **AMD Ryzen 5**, care să permită rularea eficientă a algoritmilor de procesare a imaginii și a video-urilor.
* CPU-ul va trebui să aibă cel puțin 4 nuclee pentru a asigura un proces de calcul paralel rapid, mai ales atunci când se procesează fișiere video de calitate înaltă.

**Placă grafică (GPU)**:

* Pentru procesarea rapidă a video-urilor și rularea modelelor de învățare automată, sistemul va necesita un **GPU dedicat**, cum ar fi **NVIDIA GTX 1060** sau o variantă mai performantă (de exemplu, **RTX 2060**). GPU-ul va accelera procesarea video-urilor și va îmbunătăți semnificativ viteza de rulare a rețelelor neuronale YOLO.
* GPU-ul va permite procesarea paralelă a datelor, esențială pentru detecția rapidă a jucătorilor și mingii în fiecare cadru video.

**Memorie RAM**:

* Sistemul va necesita cel puțin **8 GB de RAM** pentru a asigura o rulare fluidă a aplicației, având în vedere că procesarea video-urilor și calculul statisticilor necesită stocarea temporară a datelor.
* În funcție de complexitatea videoclipurilor procesate, utilizarea mai multor module RAM (16 GB) ar fi recomandată pentru a îmbunătăți performanța.

**Stocare**:

* Este recomandat ca sistemul să utilizeze un **SSD de 50 GB** sau mai mult pentru stocarea rapidă a fișierelor video și a datelor generate. SSD-urile sunt mult mai rapide decât hard disk-urile tradiționale (HDD), iar această performanță rapidă va asigura încărcarea și procesarea fișierelor video într-un timp scurt.

### Proiectare software de detaliu

Python este limbajul principal ales datorită ecosistemului său vast pentru procesarea imaginilor și a datelor. Este un limbaj flexibil, ușor de utilizat și foarte bine documentat, ceea ce îl face ideal pentru acest tip de aplicație.

* OpenCV: Va fi utilizat pentru încărcarea, manipularea și procesarea cadrelor video. OpenCV va fi responsabil pentru aplicarea algoritmilor de detecție a jucătorilor și a mingii folosind modelele YOLOv5 și YOLOv8.
* NumPy: Va fi folosit pentru manipularea matricelelor și pentru efectuarea unor calcule rapide pe datele numerice extrase din videoclipuri. NumPy va fi crucial pentru efectuarea operațiunilor matematice, cum ar fi calculul vitezei mingii și al distanței parcurse de jucători.
* Pandas: Va fi utilizat pentru gestionarea și analiza datelor statistice, cum ar fi viteza mingii și distanța parcursă de jucători. Pandas este extrem de util pentru manipularea datelor în structuri de date precum DataFrame-uri și va ajuta la organizarea rezultatelor într-un format ușor de analizat și vizualizat.

Sistemul va fi împărțit în module bine definite pentru a asigura o arhitectură modulară și ușor de întreținut:

* Modulul de procesare video: Va gestiona încărcarea și procesarea videoclipului. Va aplica modelele de detecție (YOLOv5 și YOLOv8) pentru a identifica jucătorii și mingea în fiecare cadru al videoclipului.
* Modulul de calcul al statisticilor: Va prelua coordonatele detectate ale jucătorilor și ale mingii și va calcula statistici, cum ar fi viteza mingii și distanța parcursă de jucători, folosind NumPy pentru calcule matematice rapide.
* Modulul de vizualizare grafică: Va desena terenul de tenis și va actualiza mișcările jucătorilor și mingii pe terenul grafic, folosind OpenCV pentru a actualiza fiecare cadru și a vizualiza statisticile calculate.

**Optimizing Performance:**

Pentru a asigura o performanță optimă:

* NumPy va fi folosit pentru a accelera calculele matriceale necesare pentru determinarea vitezei mingii și a distanței parcurse de jucători.
* Pandas va organiza eficient datele statistice și va ajuta la vizualizarea lor clară și concisă.
* În ceea ce privește vizualizarea grafică, OpenCV va fi folosit pentru a actualiza terenul grafic rapid, fără a depinde de biblioteci externe suplimentare.

### Proiectare de detaliu pentru performanța sistemului

Performanța sistemului este esențială, mai ales că procesarea video-urilor în timp real și calculul rapid al statisticilor (viteza mingii, distanța parcursă de jucători) necesită o utilizare eficientă a resurselor hardware și software.

* Optimizarea performanței hardware:
  + Utilizarea unui GPU dedicat este crucială pentru a accelera procesarea video-urilor și pentru a susține rularea modelelor de învățare automată. GPU-ul va reduce semnificativ timpul necesar pentru detecția obiectelor în video.
  + Memoria RAM va fi gestionată eficient pentru a preveni consumul excesiv de resurse atunci când mai multe procese rulează simultan.
* Optimizarea performanței software:
* Paralelizarea procesării video: Vom utiliza tehnici de paralelizare pentru a procesa cadrele video în paralel, folosind thread-uri separate pentru fiecare cadru, astfel încât procesarea să fie cât mai rapidă.
* Algoritmi de optimizare pentru viteza mingii și distanța parcursă: Calculul acestor statistici va fi realizat folosind formule eficiente din punct de vedere al resurselor. Algoritmii vor fi optimizați pentru a reduce timpii de calcul.
* Testare de performanță:
  + Sistemul va fi testat pentru a asigura că poate procesa videoclipuri de 10 secunde (cu 30 fps) într-un timp rezonabil, fără întârzieri semnificative.
  + Teste de încărcare vor fi efectuate pentru a verifica cum sistemul gestionează fișierele video mari și procesele paralele.

### Proiectare detaliată a comunicațiilor interne (între componente)

Comunicarea internă între componentele software va fi realizată prin variabile partajate și apeluri de funcții interne, fără a folosi API-uri externe. Fiecare componentă va fi responsabilă pentru o parte a procesului global, iar fluxul de date între ele va fi realizat într-un mod eficient pentru a asigura performanța sistemului.

**Comunicarea între module:**

* Modulul de procesare video va extrage coordonatele jucătorilor și mingii și le va trimite către modulul de calcul al statisticilor. Datele vor fi transmise prin variabile partajate care conțin pozițiile detectate ale obiectelor.
* Modulul de calcul al statisticilor va calcula viteza mingii și distanța parcursă de jucători pe baza coordonatelor și va transmite aceste statistici către modulul de vizualizare grafică.
* Modulul de vizualizare grafică va actualiza terenul grafic și va afișa statisticile, oferind feedback în timp real utilizatorului.