UNIVERSITATEA POLITEHNICA BUCUREȘTI

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

DEPARTAMENTUL CALCULATOARE

|  |  |
| --- | --- |
| upb | cs |

PROIECT DE DIPLOMĂ

Paintball Coop - joc multiplayer pentru ceasul Fitbit

Alexandru Udriștioiu

**Coordonator științific:**

Sl.dr.ing. Emilian Rădoi

Ing. Victor Gradinescu

**BUCUREŞTI**

2022

**CUPRINS**

[Sinopsis 2](#_Toc107075637)

[Abstract 2](#_Toc107075638)

[Mulțumiri 4](#_Toc107075639)

[1 Introducere 5](#_Toc107075640)

[1.1 Context 5](#_Toc107075641)

[1.2 Problema 5](#_Toc107075642)

[1.3 Obiective 5](#_Toc107075643)

[1.4 Structura lucrării 5](#_Toc107075644)

[2 Studiu de piață 7](#_Toc107075645)

[3 Analiza și specificarea cerințelor 11](#_Toc107075646)

[4 Soluția propusă 12](#_Toc107075647)

[5 Detalii de implementare 19](#_Toc107075648)

[5.1 Device 21](#_Toc107075649)

[5.1.1 Join Online Game 21](#_Toc107075650)

[5.1.2 Create Private Game 21](#_Toc107075651)

[5.1.3 Join Private Game 21](#_Toc107075652)

[5.1.4 Statistics 21](#_Toc107075653)

[5.1.5 Options 22](#_Toc107075654)

[5.1.6 Gameplay 22](#_Toc107075655)

[5.2 Companion 22](#_Toc107075656)

[6 Evaluarea rezultatelor 26](#_Toc107075657)

[7 Concluzii 27](#_Toc107075658)

[8 Bibliografie 28](#_Toc107075659)

[9 Anexe 29](#_Toc107075660)

# Sinopsis

Proiectul constă într-un joc de Paintball de tip multiplayer(pe echipe : aplicație ceas + mobil) pe smartwatch-ul Fitbit. Scopul jocului este de a-ti elimina oponenții. Partea de multiplayer va fi realizata folosind o baza de date menită pentru colectarea de date în timp real.

Smartwatch-urile au devenit din ce în ce mai complexe prin prisma inovațiilor aduse la nivel hardware. Acesta este unul din motivele pentru care mulți tineri au decis să achiziționeze un asemenea dispozitiv. Ceasul celor de la Fitbit este menit pentru monitorizarea sănătății și a exercițiilor fizice, dar are și o parte de jocuri singleplayer. Proiectul meu vine în ajutor pe aceasta parte ,fiind o aplicație în care participa mai mulți utilizatori în același timp.

Am construit o aplicație cu o conexiune stabilă care poate fi jucat în 2-4 oameni. Aceasta are un meniu cu o interfață prietenoasă și ușor de folosit prin care fiecare utilizator poate fie să intre într-un joc public sau privat(dacă dorește sa joace cu anumite persoane),fie sa își personalizeze aplicația după bunul plac sau sa își vadă parcursul în joc, aplicația având o parte de statistici. Jocul este performant și optim având un consum de resurse minim, bateria consumându-se cu aproximativ 1% la 4 minute de folosire. Când vorbim de consumul memoriei, aplicația mea are o dimensiune de 512KB.

# Abstract

The project consists of a multiplayer Paintball game on the Fitbit smartwatch. The object of the game is to eliminate your opponents. The multiplayer counterpart is realized using a database designed to collect real-time data.

Smartwatches have become increasingly complex in terms of hardware innovations. This is one of the reasons why many young people have decided to buy one. Fitbit smartwatch is designed to monitor health and fitness , but also includes a few singleplayer games. My project helps on this side, being an application in which several users participate at the same time.

I built an application with a stable connection that can be played în 2-4 people. I have a menu with a friendly and easy to use interface where each user can enter a public or private game (if he wants to play with certain people),  personalize their application to their liking or to see the course of their games  having  statistics calculated and showcased on companion UI provided by the app.The game is efficient and optimized with minimum resource consumption, battery being consumed by about 1% every 4 minutes of use.When we speak about memory usage,my app has an install size of 512KB.

# Mulțumiri

Dan Tudose

Laura Ruse

Cătalin Ghenea

# Introducere

## Context

Proiectul ales de mine este un joc de Paintball multiplayer pe smartwatch-ul Fitbit.

Am dorit sa aleg o tema care folosește anumite tehnologii atinse în cadrul facultății. Am avut 2 laboratoare realizate pe acest ceas și am decis sa continui ceea ce am învățat în dezvoltarea unui joc, asta fiind una dintre pasiunile mele.

Consider ca aplicațiile prin care interacționează oamenii sunt de succes, iar această latură nu există momentan în galeria Fitbit. Totodată această gamă largă de smartwatch-uri a acaparat atenția tinerilor de azi prin puterea de calcul al acestor dispozitive și prin simplitatea utilizării lor.

## Problema

Au fost adăugate în fiecare an diferite funcționalități la API-ul public dezvoltat și publicat de către cei de la Fitbit, precum și conexiunea la internet pe aplicația mobilă corespunzătoare. Totuși există o lipsă de folosire a acestora pe piață, nicio aplicație având aceasta componentă de multiplayer.

## Obiective

Prin acest proiect aș dori să dezvolt un joc distractiv, de tip shooter care sa devină una din activitățile preferate ale fiecărui grup de oameni pasionați de jocuri. În acest sens am, pe lângă implementarea jocului multiplayer(la care este nevoie de ambele părți, atât aplicația pe ceas cât și cea pe mobil),2 meniuri care mă vor ajuta în succesul proiectului : cel de opțiuni prin care se poate personaliza aplicația și cel de statistici prin care se poate vedea cum te descurci și prin care poți sa te compari raportat la prietenii tai.

În cadrul rezolvării proiectului evoluția este dată de mai mulți factori precum : conexiunea la internet, disponibilitatea bazei de date Firebase, latența trimiterii și a primirii mesajelor dintre aplicația pe ceas și cea de telefon, cât și dintre baza de date și companion.

## Structura lucrării

În partea de analiză și specificarea cerințelor voi vorbi despre necesitatea pe piață a unei astfel de aplicație și voi detalia funcționalitățile propuse împreună cu o evaluare a acestora.

În secțiunea de abordări existente voi face o comparație între soluțiile propuse pe piață și proiectul meu. Totodată voi menționa și exemplifica tehnologiile folosite urmând să motivez și alegerile făcute.

Voi creiona fiecare aspect tehnic legat de acest joc și voi detalia la nivel de utilizator funcționalitățile adăugate când voi vorbi de soluția propusă.

În subcapitolul următor voi vorbi despre implementare explicând ce, cum și unde am folosit.

Un paragraf în care fiecare dintre secțiunile următoare este prezentată în 1-2 fraze, punând accentul pe elementele cele mai semnificative din fiecare secțiune.

În penultima parte voi analiza toate cerințele produsului din punct de vedere al securității, al folosirii resurselor și din punct de vedere al utilizatorilor.

La nivel de concluzii voi prezenta și argumenta relevanța proiectului în domeniu împreună cu rezultatele obținute în urma evaluării folosind informațiile menționate anterior.

# Studiu de piață

Pe piață nu există un joc de acest tip făcut în cadrul dispozitivelor Fitbit. Există doar aplicații din afara acestui mediu închis ce folosesc date din ceas/profilul unui purtător de ceas pentru o aplicație de tip joc multiplayer. Cumva aceasta ramură a departamentului IT nu a fost încă exploatată decât la nivelul unui joc distractiv, dar de tip single player. Această concluzie a fost trasată în urma analizei pieței. Mai departe, nici celelalte smartwatch-uri nu dețin vreo aplicație pe această zonă. Totuși m-am inspirat dintr-un joc pentru partea de implementare a jocului, anume Wild West, creat de LR Styles([1]).

Au fost făcute multe studii pe baza monitorizării sănătății și pe ceea ce reușește un smartwatch să facă prin colectarea acestor date. De exemplu, un studiu de caz([2]) demonstrează cum măsurătorile făcute de pe un ceas Fitbit au reușit să ajute utilizatorii în îmbunătățirea activității fizice.

Comunicând cu diverși clienți mi s-a confirmat faptul ca cele mai de succes jocuri în diverse compartimente cum ar fi PC sau mobile sunt cele multiplayer deoarece se dezvoltă o competiție între mai mulți jucători de oriunde de pe glob, scopul nefiind doar sa te depășești tu pe tine însuți. Faptul că acest segment este momentan existent pe piață Fitbit a fost unul din motivele care m-au influențat sa aleg acest proiect de licență. Totodată am fost interesat ce au jucat utilizatorii pe un ceas smartwatch și motivele pentru care au ales să facă acest lucru pe un asemenea dispozitiv în locul unui telefon. O bună parte s-au referit la faptul că au dorit să nu consume bateria telefonului, ceasul fiind un înlocuitor destul de performant.

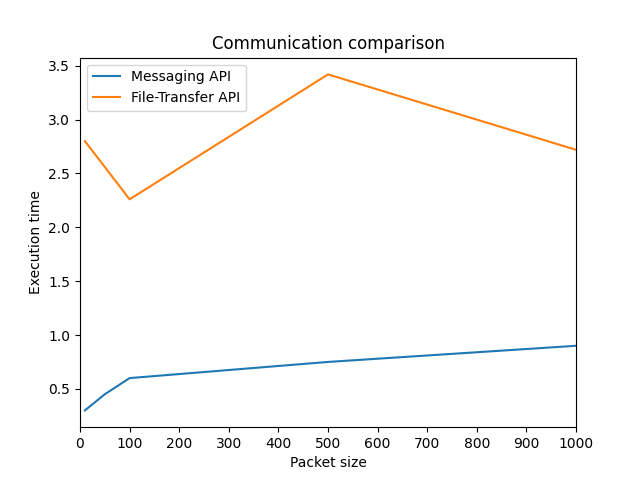
În general, un joc multiplayer este caracterizat de latență trimiterii și a primirii mesajelor între jucători(în cazul de față între fiecare jucător în parte și baza de date Firebase) deoarece utilizatorii nu doresc sa aștepte un anumit timp la fiecare acțiune realizată. O altă particularitate este dată de disponibilitatea atât a bazei de date, cât și a internetului, două componente necesare în cadrul jocului.

Tehnologiile folosite în lucrare sunt următoarele(Wikipedia) :

- **Fitbit SDK** – kit public pentru dezvoltarea atât a aplicațiilor cât și a fețelor de ceas în cadrul smartwatch-ului Fitbit. Menționez următoarele funcționalități folosite în cadrul proiectului : Document API, Storage API, Power API, System API, Geolocation API, Messaging API, Settings API, Clock API, Device API, Web API(aici am folosit XMLHttpRequest pentru cererile HTTP)

Voi detalia mai jos fiecare API folosit :

* **Power API** – folosit pentru a obține detalii despre bateria ceasului
* **System API** – folosit pentru a monitoriza memoria ceasului.
* **Document API** – folosit pentru a interacționa cu elementele create în fișierul view pe ceas
* **Storage API** – folosit pentru a prelua datele de pe UI telefonului precum câmpuri text sau alegeri făcute(ColorPicker) ,cât și pentru a le actualiza
* **Geolocation API** – folosit în a poziționa fiecare jucător pe harta în funcție de localizarea telefonului din mediul real(este singura funcționalitate care se ocupa de localizare)
* **Clock API** – folosit pentru a măsura timpul petrecut de fiecare utilizator în cadrul aplicației, funcționalitate necesara pentru partea de statistici.
* **Messaging API** – tehnologie utilizata pentru a transmite mesaje și pentru comunicare între ceas și telefon. Alte tehnologii ce ar putea fi folosite, File-Transfer API prin care putem transmite informații prin fișiere între cele 2 device-uri. Este vitală folosirea acestei funcționalități deoarece prin aceasta putem comunica între device-uri. După cum se vede și în figura 1 , API folosit este și mult  mai rapid ca timp de execuție.



Figură 1 Messaging vs File-Transfer

* **Device API** – folosit pentru a prelua dimensiunea dispozitivului . Geolocația îmi poate furniza poziția telefonului în perechea (latitudine, longitudine) și am nevoie de mărimea ceasului(mai exact înălțimea și lățimea) pentru a putea mapa aceasta pereche de coordonate în (x,y)
* **XMLHTTPREQUEST** – folosit pentru a realiza cereri de tip HTTP către baza de date Firebase pentru metodele de GET(citire date),POST(scriere date),PUT(modificare date) și DELETE(ștergere date).Partea cea mai importantă din aplicație, mai exact cea de multiplayer, este realizată folosind această tehnologie. Aș fi putut folosi și Web Socket API prin care comunicarea se face printr-un server(toată logica aplicației ar trebui mutată pe server și nu aș mai folosi ceasul).
* **Settings API** – aplicația de mobil funcționează doar dacă există aceasta parte de “setări”. Aici am putut sa realizez multe dintre cerințele furnizate de meniul aplicației pe ceas. În plus, din motive de securitate, aici am putut sa pun la dispoziție Google Oauth2 prin care fiecare utilizator se autentifică folosind credențialele necesare care mă vor ajuta sa obțin autorizația pentru a accesa datele acestora din Fitbit.
* **Web API** – având nevoie de informații din profilul public al fiecărui utilizator(pe baza de autoritate furnizata mai sus) trebuie sa pot face cereri de tip GET pe această pagina.

- **Baza de date Realtime Firebase** – baza de date de tip NoSQL, localizată în cloud, prin care noi, dezvoltatorii, putem stoca informații foarte simplu și foarte ușor. Un avantaj oferit este acela ca datele se actualizează în timp real. Am ales aceasta baza de date deoarece este mult mai potrivită pentru tema aleasă de mine(fiind un joc multiplayer datele trebuie sa fie modificate imediat, costul este redus deoarece nu se trimit enorm de multe date).

Ca și alternative aș fi putut folosi baza de date DynamoDB a celor de la AWS(doar ca aceasta se mulează pe aplicații cu cantități mare de date și unde sunt necesare operații de procesare complexe ,iar aici nu este cazul) sau MongoDB(aceasta baza de date se pretează mai bine pe zona de Big Data unde se analizează seturi de date foarte mari).

**O bază de date NoSQL** este un tip de baze de date folosit în mare parte în Big Data și în aplicațiile web. Putem stoca date sub orice formă, nu trebuie sa aibă o structură fixă(cum se întâmplă în SQL).([4],[5])

**Cloud computing** este o tehnologie formată dintr-un grup de servicii conectate între ele prin Internet. În esență, acestea sunt ținute la distanță printr-un furnizor de servicii de cloud. Aceasta tehnologie ne oferă spor de performanță și de cost(folosim servere aflate la distanta pentru stocare și nu mai cumpărăm echipamente menite cu acest scop).([6],[7])

- **Javascript** – este un limbaj de programare orientat obiect folosit mai ales pentru crearea aplicațiilor web. Codul rezultat este rulat de către browser. În cazul de față este folosită doar o parte din acest limbaj de programare, în cadrul produselor marca Fitbit având un mediu închis pentru dezvoltarea aplicațiilor.([8],[9])

-JSX – extensie care ne ajuta la scrierea un cod asemenea celui HTML ( pentru a crea interfața aplicației mobile)

# Analiza și specificarea cerințelor

În urma analizei pieței am ajuns la următoarele funcționalități:

* Utilizatorii își pot personaliza aplicația după bunul plac
* Utilizatorii își pot vedea parcursul în joc menționând : timpul petrecut în aplicație, numărul de jocuri desfășurate, procentajul de câștig și procentajul de gloanțe nimerite. Am observat ca o mare majoritate oamenilor angrenați în jocurile de tip multiplayer vor sa vadă cum se descurcă și/sau sa își compare parcursul cu prietenii săi.
* Utilizatorii pot sa creeze un joc privat astfel încât să intre cu prietenii sa se distreze. Cumva trebuie sa existe o opțiune prin care sa alegi cu cine vrei sa joci fără sa fie totul impus de către aplicație.
* Utilizatorii pot să intre într-un joc multiplayer public dintr-o lista generata deja
* Partea de joc este redata folosind un mediu specific, jucătorii sunt reprezentați prin imagini ce includ echipamentul specific unui jucător de paintball, anume : cască de protecție, armură și un pistol cu vopsea),iar glonțul este reprezentat de o imagine în care se vede vopseaua împrăștiată asemenea unei arme din realitate.

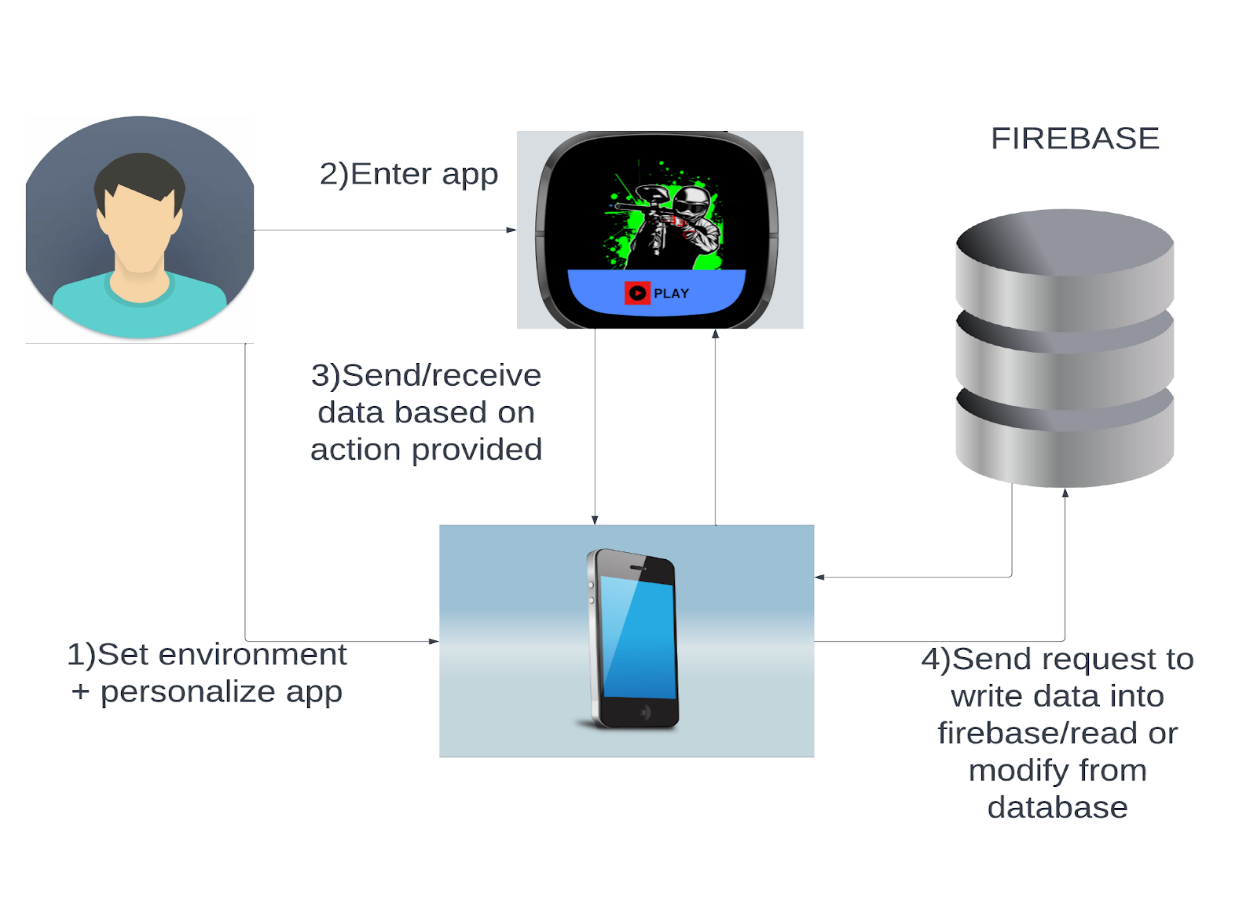
În cadrul dezvoltării cerințelor am luat la cunoștință și anumite probleme ale ceasurilor. De exemplu, proiectarea unui joc [3] pe ceas înseamnă ca avem o mobilitate mai scăzută (având in vedere ca ecranul unui ceas nu este foarte mare) și dacă vorbim de trecerea unui joc făcut creat pe calculator sau mobil pe ceas trebuie făcute multe modificări deoarece nu sunt aceleași capabilități.

Inițial, prototipul aplicației conținea și alte funcționalități precum : existența unei liste de jucători împărțită în lista de prieteni(cu care se putea interacționa prin trimitere de mesaje sau prin ștergerea/blocarea acestuia + la nivel de statistici să poți compara cu aceștia) și lista de jucători recent întâlniți(pe care ii poți adăuga în lista de prieteni).În urma interviurilor am decis ca aceasta idee ar complica foarte mult aplicația și că ar putea fi adăugată într-un viitor apropiat. De asemenea, prin feedback-ul provenit de la clienți ,am validat fiecare funcționalitate în parte având o parte de reevaluare a acestora punând accentul pe avantaje și dezavantaje. Fiind limitat de capabilitățile implementării pe aplicația de mobil am analizat încă o data cerințele funcționale și am decis sa le păstrez schimbând doar modul de funcționare(datele sunt completate în aplicația de mobil și actualizarea acestora în cadrul aplicației făcându-se de pe smartwatch)

Fiind pasionat de jocuri, am dorit sa creez o astfel de aplicație în mediul Fitbit, mai ales văzând că există doar jocuri single-player pe piață.

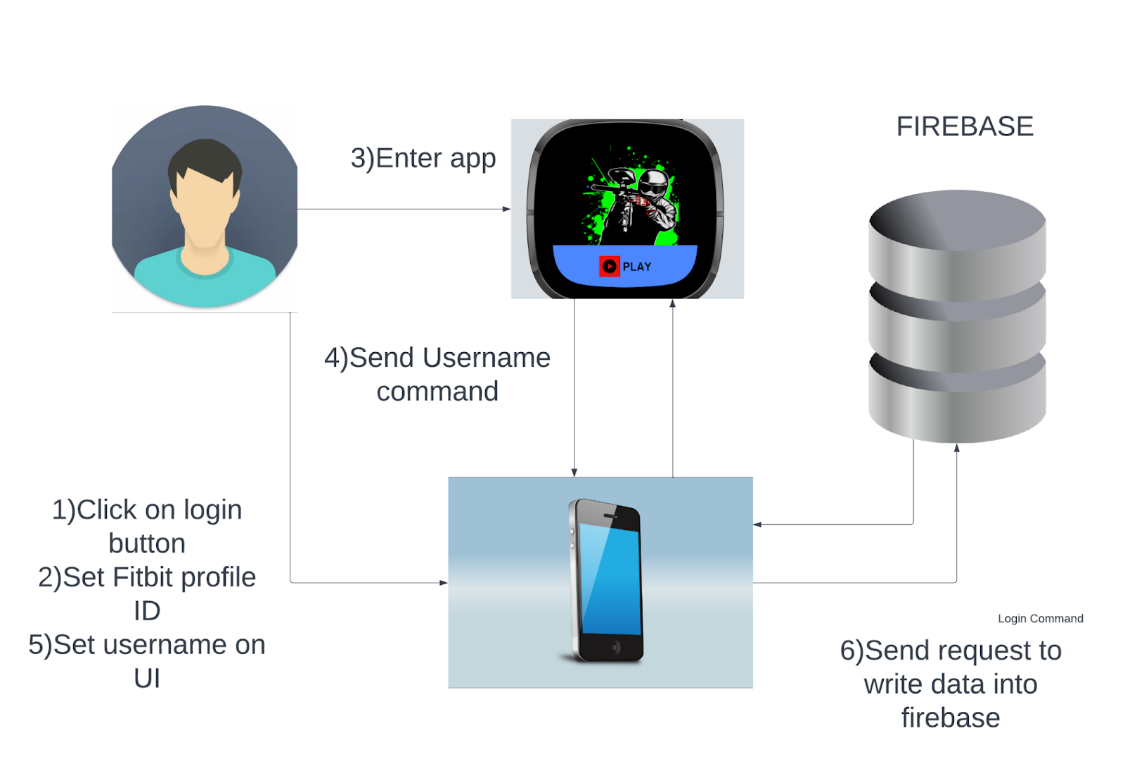
# Soluția propusă

Structura aplicației este descrisă în figura 2. Fiecare utilizator în parte interacționează cu elementele din cadrul ceasului folosind aplicația mobilă trimițând o comandă pentru fiecare acțiune în parte. În aplicația de pe telefon se vor modela sau accesa datele necesare. Totodată prin aplicația mobilă, având conexiune la internet, se va face corelarea și cu baza de date Firebase pentru adăugarea utilizatorilor, a camerelor de jocuri ,atât private cât și publice, și pentru implementarea jocului multiplayer(poza pentru avatar a fost luată de aici [10],cea cu telefonul de aici [11] și cea cu baza de date de aici [12]).



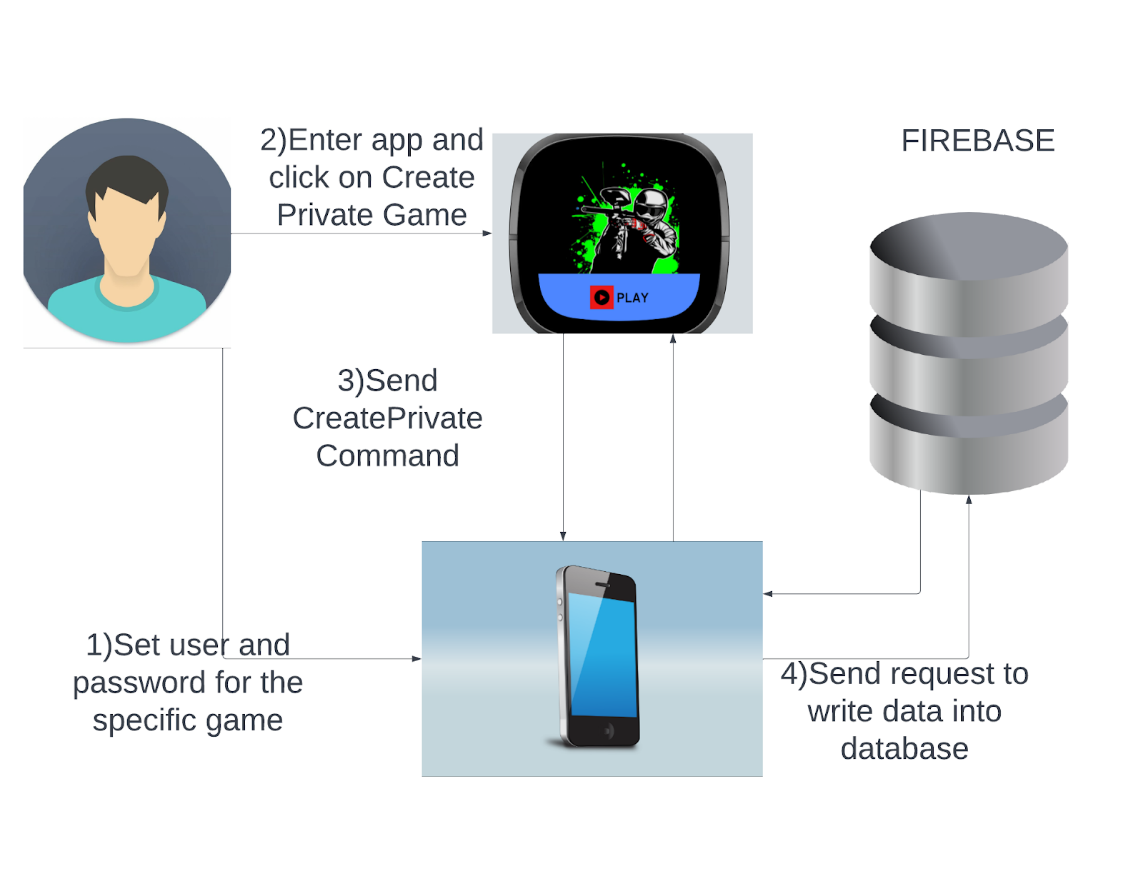
Figură 2 App Flow

Figura 3 reprezintă autentificarea utilizatorului folosind Google OAUTH2.La apăsarea butonului de login se va accesa un token de autorizație folosit pentru a accesa profilul public al fiecărui utilizator printr-o cerere HTTP.Utilizatorul setează pe aplicația mobilă  id-ul profilului Fitbit pentru preluarea datelor de pe profilul public înainte.



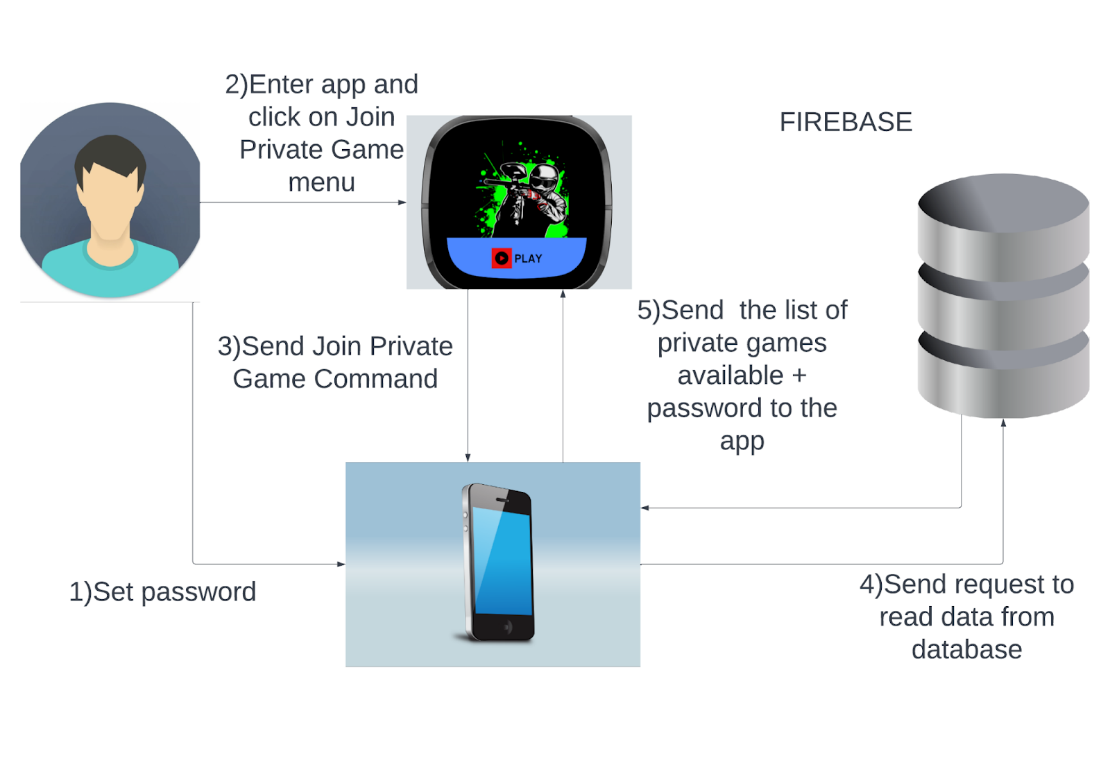
Figură 3 Google OAUTH2

În figura 4 este afișat cum funcționează meniul de Create Private Game. Utilizatorul setează un nume și o parola astfel fiind creată o nouă intrare ce va fi adăugată în baza de date.



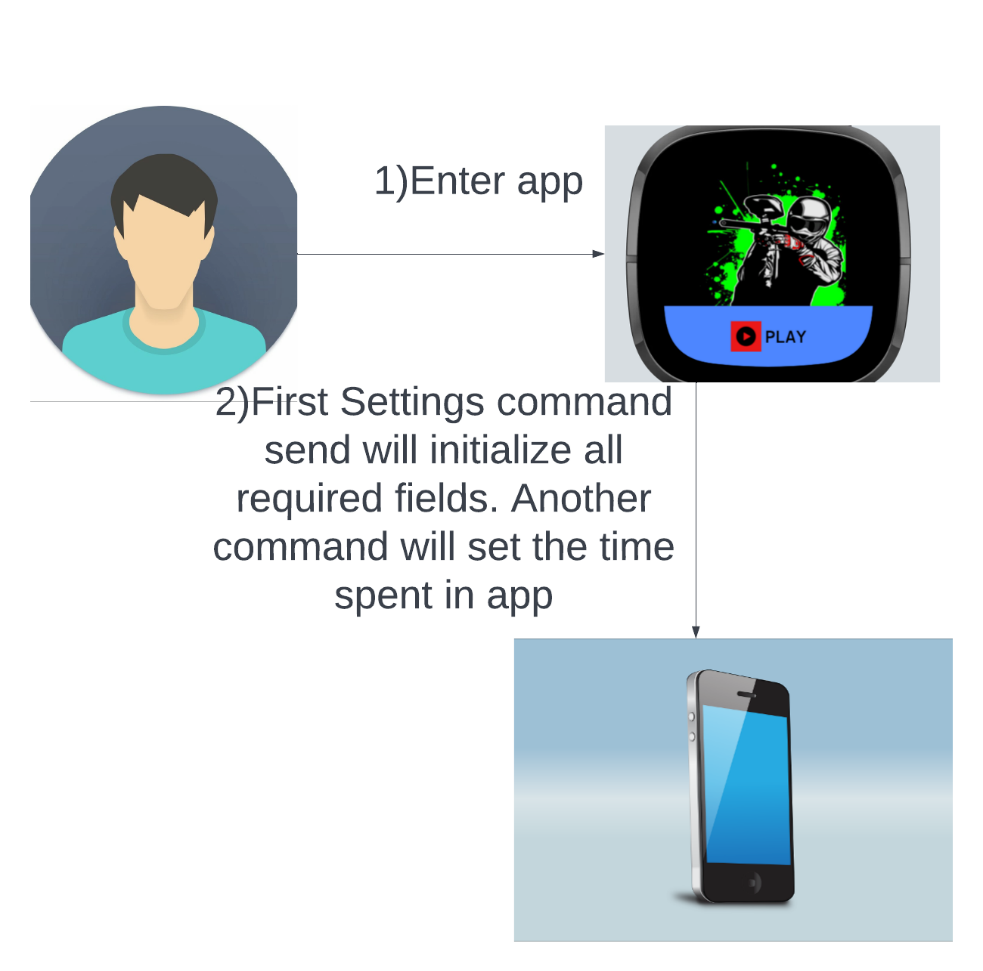
Figură 4 Create Private Game

La fel funcționează și meniul de Join Private Game după cum se vede în figura 5. Utilizatorul setează un singur câmp, parola, ce fi folosit la urmă pentru intrarea într-o camera de joc privată. Ambii jucători vor intra în același timp, iar ulterior cine vrea sa mai între în acest meci nu va avea acces deoarece limita de jucători a fost atinsă. Aceștia se vor aștepta reciproc pentru o buna desfășurare.



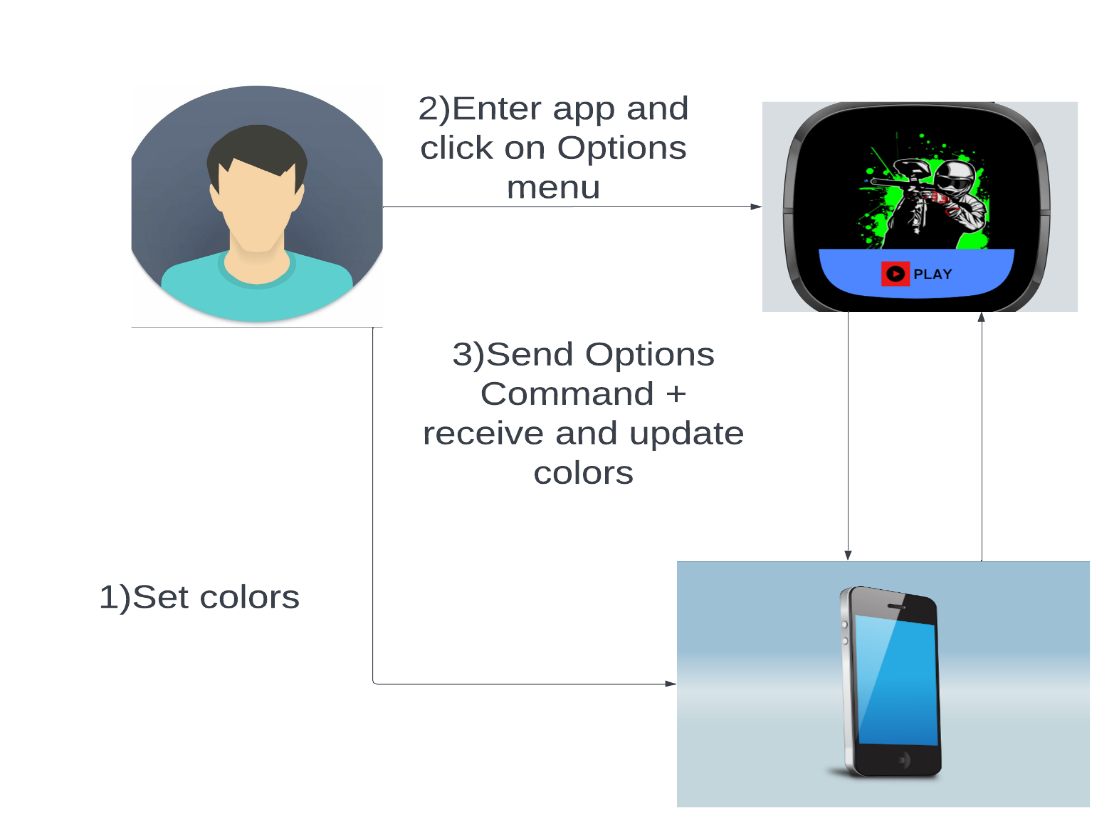
Figură 5 Join Private Game

În figura  6 este reprezentat meniul de Statistics unde utilizatorul își vede parcursul în joc pe baza a ceea ce a realizat și cât timp și-a petrecut în cadrul aplicației. Timpul petrecut în joc este măsurat de când s-a apăsat butonul de play din primul ecran și până se accesează cel de Statistics unde se va trimite mai departe către Companion.



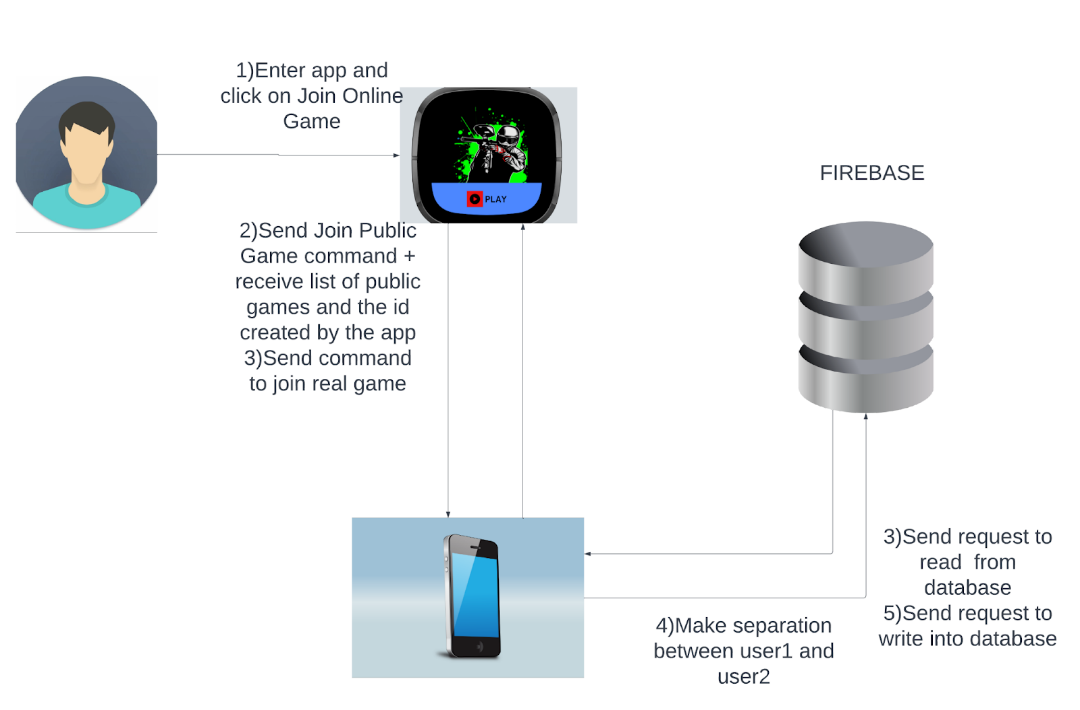
Figură 6 Statistics

În figura 7 este reprezentat meniul de Options unde fiecare utilizator alege culorile jucătorilor marcate pe hartă ca fiind pozițiile acestora în cadrul jocului. În figura 6 este reprezentat meniul de Join Online Game unde utilizatorii intra în camere aleatorii și așteaptă un oponent(pentru camera privată jucătorii intra pe măsură ce selectează meniul și camera aleasă, iar aceștia vor intra direct în joc se vor aștepta unul pe celălalt pentru a putea începe).



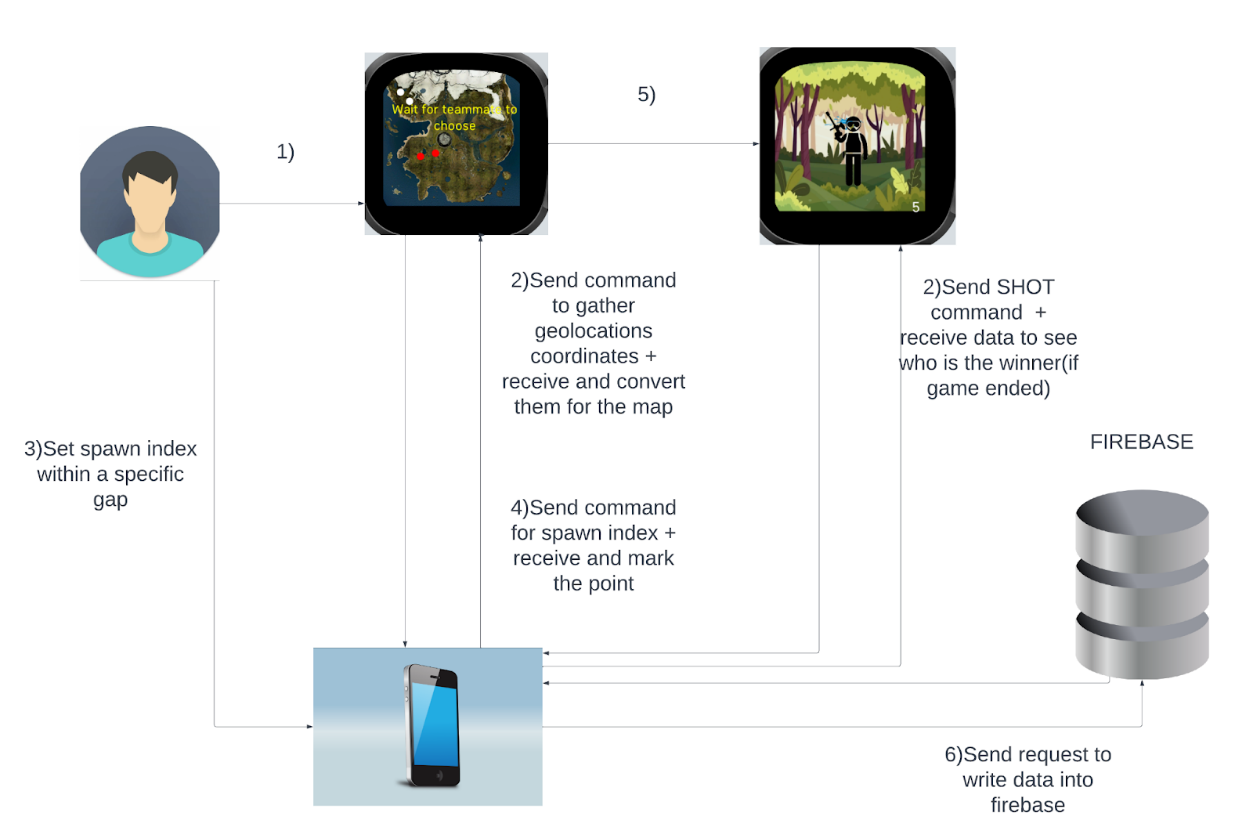
Figură 7 Options

În figura 8 este reprezentat meniul de Join Online Game unde utilizatorii intra în camere aleatorii şi așteaptă un oponent(pentru camera privată jucătorii intra pe măsură ce selectează meniul și camera aleasă direct în joc și se vor aștepta unul pe celălalt pentru a putea începe).Pentru acest timp de așteptare se trece prin pașii 2 și 3(folosind același procedeu dar o altă comandă pentru a nu avea conflict de date) pana când expira perioada de timp sau încă un jucător a intrat în sesiunea respectivă. La jocurile private nu se mai face aceasta bucla cu pașii 2-3 și se intra direct în joc.



Figură 8 Matchmaking

În figura 9 este descrisă implementarea jocului la nivel de interacțiuni cu baza de date. Prima oară vor intra cei doi jucători, fie folosind o parola pentru jocurile private, fie pe baza unui id aleator setat în aplicație pe baza căruia i se va căuta sesiunea corespunzătoare în care va intra dacă vorbim despre jocurile publice.



Figură 9 Gameplay

La nivelul jocurilor private este setat un câmp în baza de date prin care se verifică dacă numărul maxim de jucători într-o camera a fost atins, caz în care oricine ar încerca să între în acel joc va primi un mesaj ca jocul este plinul nivelul jocurilor publice pur și simplu aceste camere vor fi evitate și se va seta un nou id aleator.

Pentru primul tip de joc nu există un timp de așteptare între cei doi jucători, aceștia vor intra automat și vor cădea de comun acord pentru începerea acestuia. Pentru cel de-al doilea tip este implementat un sistem în care primul jucător intra și așteaptă intrarea celui de-al doilea pentru o perioada de timp. Dacă perioada de timp a expirat atunci jucătorul este introdus înapoi în meniul de joc urmând să încerce din nou sa se conecteze. Altfel ,când intra cel de-al doilea jucător, este semnalat și primul și astfel jocul poate sa înceapă.

La începutul jocului se va vedea harta cu 2 poziții standard de început ale inamicilor și 2 poziții construite folosind geolocația de pe telefon ale echipei jucătorului. După aceea există un timp de 3 secunde în care utilizatorul alege poziția 1 sau 2 pe aplicația mobilă după care se va intra în jocul propriu zis.

Fiecare jucător va vedea o lume, un jucător în care va trage și în dreapta jos numărul de gloanțe(în număr de 10 care se va reseta de fiecare data pana când jocul se va încheia).Acesta apasă în locul în care vrea sa tragă dacă a nimerit zona indicată aleator de joc atunci datele vor fi scrise în baza de date, altfel dacă nu a ghicit zona nu se va întâmpla nimic. Când viața unuia dintre jucători a ajuns la o valoare mai mica sau egal cu 0 atunci se va trimite un mesaj corespunzător jucătorului că a câștigat, iar ulterior după ce reușește și cel de-al doilea jucător să nimerească un glonț că a pierdut. Apoi se vor șterge intrările aferente în baza de date.

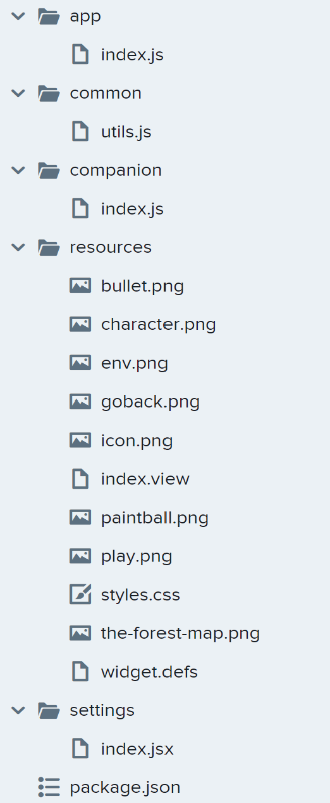
Celelalte metrici din partea de statistici vor fi calculate pe baza gameplay-ului având în vedere numărul de gloanțe trase(pentru a calcula rata de guessed shot ) și numărul de meciuri jucate(pentru a calcula procentajul de victorii).Numărul de gloanțe este contorizat de fiecare data când se trimite comandă de SHOT, iar numărul de meciuri de fiecare data când se termina.

# Detalii de implementare

O problemă generală a fost sa mă asigur ca fac separație între comenzi atât pe device cât și pe mobil deoarece dacă am 2 mesaje pe care primesc aceleași variabile ca și nume, se va intra pe fiecare listener.

În figura 10 se observă structura aplicației formată din următoarele fișiere :

1. app – aici avem codul scris pentru smartwatch-ul Fitbit ce include logica aplicației împreună cu funcționalitățile menționate anterior(folosind toate API-urile precizate în afară de Geolocation și fără partea de internet)
2. common – aici avem partea de variabile globale folosită în ambele părți(telefon și ceas),cât și funcții comune.
3. companion – aici avem codul scris pentru aplicația mobilă : partea de primire a mesajelor, corelarea cu baza de date Firebase și API-ul de geolocație
4. resources – aici avem pozele folosite în cadrul aplicației pentru a reda peisajul în care se va desfășura jocul, personajele, gloanțele + un fișier view unde sunt puse toate elementele folosite + un fișier css în care am setat diverși parametrii asupra unor componente precum butoanele din joc + un fișier defs folosit pentru a importa anumite elemente spre a fi folosite în aplicația de pe ceas cum ar fi o listă de elemente.
5. settings – aici avem partea de UI a aplicației de pe telefon având incluse câmpuri text ce vor fi completate de către fiecare utilizator în parte, partea de login folosind Google OAUTH2,campuri de text blocate spre a fi completate pe baza parcursului în cadrul aplicației și un ColorPicker pentru personalizarea jocului.
6. package.json – aici sunt incluse setările aplicației precum permisiunile acordate, SDK folosit sau tipurile de ceas pe care va funcționa aplicația.



Figură 10 Structură proiect

De menționat faptul ca jocul se desfășoară prin baza de date Firebase având structura următoare (după cum se vede și în figura 11):

1. Users – aici apar toți utilizatorii care s-au logat în aplicație
2. PublicGames – camerele de joc publice fiecare având un id
3. PrivateGames – camerele de joc private fiecare având nume și parola
4. ActivePublicGames – desfășurarea jocurilor publice adăugându-se următoarele informații : user1,hp1,user2,hp2,nume camera de joc(informații preluate din PublicGames)
5. ActivePrivateGames – structura asemănătoare cu cea de la jocurile active publice.



Figură 11 Structură baze de date

## Device

Pentru partea de ceas aplicația este construită din elementele create în fișierul view din resources. O primă funcție implementată aici a fost cea de **hideElements()** prin care am setat vizibilitatea fiecărui element la „none” urmând să apară pe ecran pe măsură ce apare un eveniment din partea utilizatorului sau făcând parte din ecranul meniului ales.

După ce se apasă butonul de play utilizatorului îi apare o lista cu meniuri fiecare cu funcționalitatea ei urmând să le descriu pe fiecare în parte. Totodată se trimite comandă USERNAME. Toate comenzile sunt trimise către aplicația mobilă, urmând acolo sa se facă o acțiune specifica.

## 5.1.1 Join Online Game

Când apasă acest buton utilizatorul i se asociază un id aleator cu sesiunea de joc la care dorește să între, prima oară verificându-se dacă acea camera este disponibilă sau nu(adică dacă au intrat deja 2 jucători),caz în care i se va genera un alt id. Dacă a reușit utilizatorul sa între sunt 2 cazuri. Primul caz în care este primul jucător, atunci acesta trebuie sa aștepte, adică se va trimite o comandă prin care se va citi baza de date actualizată și se va încerca verificarea intrării celui de-al doilea jucător pentru a putea începe jocul. Pentru a evita problema unui ciclu infinit a fost setat la început un parametru numit attempts(5) astfel încât după un timp acesta sa se întoarcă în meniul principal dacă cineva nu a intrat în sesiunea respectivă. Dacă intra pe post de al doilea jucător atunci acesta intra direct în joc, urmând ca prin verificare să între și primul după un timp.

## 5.1.2 Create Private Game

În acest meniu se trimite o comandă către telefonul mobil .Totodată se afișează și un mesaj corespunzător ca aceasta acțiune de creare se va face pe companion. La fel se întâmplă și la meniurile de Statistics și Options

## 5.1.3 Join Private Game

În al doilea tip de joc, cel privat, prima oară se primesc lista de jocuri private împreună cu parola pusă pe aplicația mobilă. Se verifica pentru fiecare camera dacă parola este cea corectă, caz în care se face adăugarea jucătorului fie ca user1 sau user2 folosind pentru fiecare un câmp creat în baza de date PrivateGames. După ce a intrat al doilea jucător va rămâne un câmp prin care se va afișa oricărui alt utilizator faptul ca acea camera de joc este plina și nu mai poate intra nimeni.

## 5.1.4 Statistics

În acest meniu se fac inițializările statisticilor afișate pe telefonul mobil când se intra prima oară. La următorul acces se va trimite către telefon timpul petrecut de utilizator în aplicație(folosind **Clock API)**.Am făcut diferența între ora la care a fost deschisă aplicația și ora la care s-a click pe butonul de Statistics.

## 5.1.5 Options

Aici se trimite o comandă către telefon prin care se vor prelua culorile alese. Aici am întâlnit o prima problema(la fel ca la Join Private Game și la Join Online Game) în care primirea mesajelor(folosind **Messaging API** atât pentru primire cât și trimitere) se face folosind un EventListener, practic așteptam un eveniment în urma căruia sa putem procesa datele recepționate. Astfel eu nu puteam sa atribui ceea ce am primit către o variabila din cadrul aplicației. Așa ca primirea datelor s-a făcut într-o funcție auxiliara ,mai apoi sa apelez funcția în care se face atribuirea.

## 5.1.6 Gameplay

Când un utilizator intra în jocul propriu a zis, prima oară se trimite o comandă prin care se preiau coordonatele de la telefon(folosind **Geolocation API)**,acestea sunt transformate din (latitudine, longitudine) în (x,y) folosind **Device API** pentru a mapa coordonatele în dimensiunile ceasului , vor fi setate pozițiile jucătorilor(atributele x și y ale elementelor aferente) și se va afișa o harta cu localizarea fiecărui jucător. Apoi se va aștepta timp de 3 secunde pentru a alege punctul de start(având de ales între cele 2 locații) ,mai apoi jucătorul va vedea poziția evidențiată schimbând culoarea din alb sau cea aleasă de pe telefon în verde. Apoi va fi afișat mediul de joc cu un jucător și în dreapta jos cu numărul de gloanțe(10).Acesta se resetează de fiecare data când se trag 10 gloanțe verificând dacă suma punctajelor obținute a depășit sau nu 100.

Am setat 6 poziții în care un jucător poate trage : cap, corp, braț stânga, braț dreapta, picior stâng, picior drept. Pentru fiecare glonț se va alege un număr aleator în intervalul [1,6] pe baza căruia se va face verificarea dacă jucătorul a țintit corect sau nu. Fiecare zona are un punctaj asociat, trimițându-se o comandă împreună cu locația, punctajul, inamicul(setat în funcție de jucător, dacă a intrat ca 1 inamicul este 2,iar dacă a intrat ca jucătorul numărul 2 inamicul este 1),o variabila care ne arată dacă zona a fost intuita sau nu(1 sau 0) și locația. În plus la jocurile publice apare și id-ul sesiunii de joc.

De fiecare data există un listener pe baza căruia se verifica câștigătorul meciului caz în care se va trimite un mesaj adecvat.

## Companion

Aplicația de pe telefon este bazata pe primire de mesaje din partea ceasului ,interacționarea cu interfața creată folosind JSX și prin realizarea de cereri HTTP utilizând **XMLHTTPRequest**. Voi descrie fiecare comandă în parte și ce se face pentru fiecare. Fiecare preluare sau modificare a elementelor se face folosind **Storage API**.O problema apărută a fost la trimiterea cererilor HTTP deoarece primeam de 2 ori răspunsul, o data pe baza comenzii OPTIONS(un fel de verificare a serverului legat de corpul cererii trimise de către mine) și a doua oară tipul dat(GET,PUT,POST,DELETE).Folosind o variabila m-am asigurat ca se executa doar o cerere.

1. **USERNAME**

La primirea acestei comenzi se va interoga baza de date a utilizatorilor pentru ca acesta sa fie adăugat. O problemă aici a fost aceea ca se creau intrări multiple ale aceluiași jucător, așa ca am fost nevoit sa fac o ștergere a intrării vechi și adăugarea celei noi de fiecare data când se primește aceasta comandă. Totodată aici se face o cerere HTTP către pagina de profil a utilizatorului folosind id dat prin telefon pentru a putea seta câmpul de utilizator pe companion. Dacă utilizatorul are setat câmpul de username pe pagina de pe Fitbit ,atunci acesta va fi adăugat. Altfel va fi pus id-ul curent.

1. **STATISTICS**

Pentru aceasta comandă sunt inițializate toate câmpurile : timp petrecut, număr gloanţe, număr jocuri, procentaj câștig, procentaj guessed shot .Mai apoi este preluat câmpul de timp și este actualizat pe baza valorii primite de la ceas. Am folosit doua funcții de conversie aici : convertTime pentru a trece din format „hh:mm:ss” în număr de secunde și convertToTime invers.

1. **OPTIONS**

Aici sunt preluate culorile alese pentru echipa sa și pentru inamici și sunt trimise înapoi către device.

1. **JOIN\_ONLINE\_GAME**

Aici se primește și locația(url-ul către care fac cereri).Dacă jocul este de tip privat(locația este 2) atunci pur și simplu se face o interogare a bazei de date(cerere de tip GET) PrivateGames(0) și se trimite înapoi lista de jocuri(ca fiind o lista cu fiecare element de tip lista de 2 elemente – numele părintelui din Firebase și un obiect ce conține informațiile acestuia) și parola pusă de utilizator pe telefon. Dacă jocul este de tip public atunci se face o verificare ca id-ul este valid(folosind baza de date ActivePublicGames), adică se verifica faptul că acea sesiune de joc nu are 2 jucători deja. Dacă are, atunci se va genera alt id și se face alta verificarea aceasta din nou. Dacă jucătorul poate intra se va trimite înapoi către ceas id-ul camerei împreună cu lista de jocuri publice(se citește baza de date PublicGames în acest caz).

1. **CREATE\_PRIVATE\_GAME**

Pentru aceasta comandă se preiau numele de utilizator și parola camerei, se creează un obiect cu acestea și este trimis mai departe spre a fi scris în baza de date corespunzătoare PrivateGames.

1. **GAME\_POSITION**

Aici se ia poziția curentă setată de poziția pe hartă a telefonului și se trimite înapoi către ceas.

1. **JOIN\_PUB**

Se creează un obiect folosind id-ul jucătorului, numele camerei primit de la ceas și viața acestuia inițializată cu 100.După ce se citesc informațiile necesare din ActivePublicGames se face verificarea tipului de utilizator pentru a fi adăugat. Totodată atunci când intră al doilea jucător se scrie încă o informație în baza de date ,mai exact în PublicGames, cum ca acea camera este plina.

1. **JOIN\_PRIV**

Aici se întâmplă același lucru ca la JOIN\_PUB doar ca interogarea se face a bazei de date ActivePrivateGames, respectiv PrivateGames.

1. **SHOT**

De fiecare data când se trage un glonț se recalculează numărul de gloanțe și rata de gloanțe nimerite având funcții de conversie din procent în număr întreg și vice versa. Dacă parametrul primit care ne indică dacă jucătorul a țintit corect sau nu este 1 doar atunci se va trimite un pachet în funcție de tipul de joc, privat sau public şi în funcție de cine a inițiat acțiunea(jucătorul 1 sau 2). Aici se face o diferență între valoarea vieții jucătorului din baza de date și cea provenită în urma jocului și se face o verificare dacă aceasta este mai mica sau egal cu 0.În acest caz există un câștigător al jocului și va fi pus un parametru în baza de date pentru a identifica acest lucru. Iar pe baza acestuia se vor trimite mesaje către ceas pentru a semnala jucătorilor încheierea meciului și întoarcerea lor în meniul de start.

**5.2.10 FLAG**

Pentru aceasta comanda se inițializă camerele, mai exact este scris un obiect în care se afla id-ul părintelui generat când a fost introdus un nou element în Firebase ,fie de tip ActivePrivateGames sau ActivePublicGames și este semnalată intrarea primului jucător(se trimite și o comandă de tip JOIN\_PUB sau JOIN\_PRIV) fie în PrivateGames sau PublicGames.

**5.2.11 GAME\_OVER**

De fiecare data când un joc s-a încheiat se recalculează numărul de jocuri jucate, procentajul de câștige asemenea se vor ștergere intrările create în bazele de date în care se desfășoară jocurile și câmpurile extra folosite pentru a face separația între utilizatori. Aici a apărut o mica problema în care se ajungea de 2 ori sa se execute comanda respectivă așa ca am folosit o variabila pentru a contoriza o singura data și a face o singura data calculul și ștergerile aferente.

**5.2.12 GAME\_CHECK**

Aici pur și simplu se recitește baza de date cu informațiile actualizate și se trimit înapoi la ceas spre verificare.

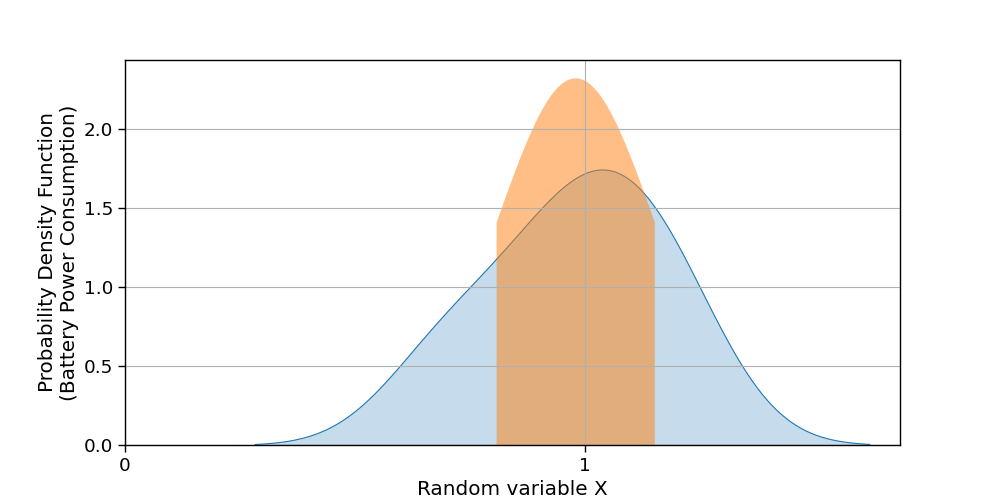
**5.2.13 GAME\_TIMEOUT**

Pentru aceasta comanda se șterge câmpul din camera asociată cu id-ul respectiv din PublicGames și se șterge camera activa deoarece aceasta nu a condus la un joc.

# Evaluarea rezultatelor

Pe baza tuturor cerințelor menționate în capitolul 3 pot spune ca aplicația merge în mod corect. La acest lucru a ajutat testarea fiecărei componente după ce a fost creată.

Legat de evaluarea cerințelor am folosit mai multe metrici pentru a trage o concluzie. În primul rând, folosind **Power API** am putut monitoriza bateria pe măsură ce aplicația rulează și sa vad cu cat scade procentajul acesteia și în cât timp. În urma a mai multor rulări succesive se observă că aplicația nu consumă foarte multe resurse, aceasta descărcându-se cu aproximativ 1% la 4 minute. Acest lucru este demonstrat şi în figura 12 unde este reprezentată funcția de probabilitate a densității pe baza consumului de baterie construit dintr-un set de 5 valori(5 rulări ale aplicației).



Figură 12 Consum baterie

Un al doilea aspect monitorizat a fost consumul memoriei unde intervine **System API**. Cu acesta am putut vedea cam cât consumă aplicația mea având în vedere că la fiecare pas se adaugă/scot obiecte ceea ce modifică și dimensiunea folosită. Cea mai mare valoare obținută(atunci când avem cele mai multe elemente pe ecran) este e 35.784 KB, ceea ce este o valoare foarte buna datorata și a imaginilor folosite(dimensiune și tip).Dimensiunea totala a aplicație este de 512 KB.

Pentru partea de networking și de comunicație între device și telefon lucrurile stau foarte bine. Baza de date Firebase are locația în Europa de Vest(mai exact Belgia) așadar timpul de răspuns este foarte bun și latenta foarte mica. Cererile HTTP către endpoint-ul bazei de date se execută într-un timp foarte scurt. Totodată pentru partea a doua, folosind **Messaging API**, fiecare mesaj este transmis și recepționat în mai puțin de o secundă, un timp care nu îngreunează aplicația.

# Concluzii

Proiectul rezultat propune un nou tip de joc adus in rândul aplicațiilor din gama Fitbit reușind sa creeze un mediu distractiv in care se pot juca mai mulți oameni împreună. Un alt aspect important este faptul ca acest joc multiplayer este realizat fără server(totul se face prin baza de date folosind cereri HTTP).

Am realizat o analiza din punct de vedere al cerințelor și am explicat ce tehnologii au fost folosite și de ce sunt mai bune decât cele existente. La nivel de implementare a fost prezentat cum s-a făcut comunicarea dintre ceas și telefon, telefon și baza de date, cât și cum comunica cei doi jucători din cadrul unei camere de joc folosind ca instrument tot baza de date.

Aceasta aplicație este performanta și optima din mai multe puncte de vedere. Prin prisma rezultatelor de mai sus se observa ca resursele sunt consumate într-o cantitate foarte mica. De asemenea conexiunea este rapidă, mesajele transmițându-se in cel mult o secunda, fie ca vorbim intre aplicația de ceas și cea de mobil, fie ca vorbim despre cererile către baza de date Firebase.

# Bibliografie

[1] https://gallery.fitbit.com/details/42fcccc4-3318-43c6-8ee7-13867e144d71

[2] [Mickael Ringeval](https://www.jmir.org/search?term=Mickael%20Ringeval&type=author&precise=true) ; [Gerit Wagner](https://www.jmir.org/search?term=Gerit%20Wagner&type=author&precise=true) ; [James Denford](https://www.jmir.org/search?term=James%20Denford&type=author&precise=true) ; [Guy Paré](https://www.jmir.org/search?term=Guy%20Par%C3%A9&type=author&precise=true) ; [Spyros Kitsiou](https://www.jmir.org/search?term=Spyros%20Kitsiou&type=author&precise=true) ,“Fitbit-Based Interventions for Healthy Lifestyle Outcomes: Systematic Review and Meta-Analysis”

2020.[Online] Available : <https://www.jmir.org/2020/10/e23954/>

([Journal of Medical Internet Research - Using Fitness Trackers and Smartwatches to Measure Physical Activity in Research: Analysis of Consumer Wrist-Worn Wearables (jmir.org)](https://www.jmir.org/2018/3/e110/),

[Health at hand: A systematic review of smart watch uses for health and wellness - ScienceDirect](https://www.sciencedirect.com/science/article/pii/S1532046416301137))

[3] Li Huangyiu, “User experience Design for Smartwatch Games”,2017.[Online] Available: https://core.ac.uk/download/pdf/132419454.pdf

[4] https://www.oracle.com/ro/database/nosql/what-is-nosql/

[5] https://www.couchbase.com/resources/why-nosql

[6] https://www.oracle.com/ro/cloud/what-is-cloud-computing/

[7] https: //uniserveit.com/blog/the-basics-of-cloud-computing

[8] https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First\_steps/What\_is\_JavaScript

[9] https://www.freecodecamp.org/news/what-is-javascript-definition-of-js/

[10] https://cdn5.vectorstock.com/i/1000x1000/51/99/icon-of-user-avatar-for-web-site-or-mobile-app-vector-3125199.jpg

[11] https://pixabay.com/vectors/mobile-phone-mobile-phone-2198770/

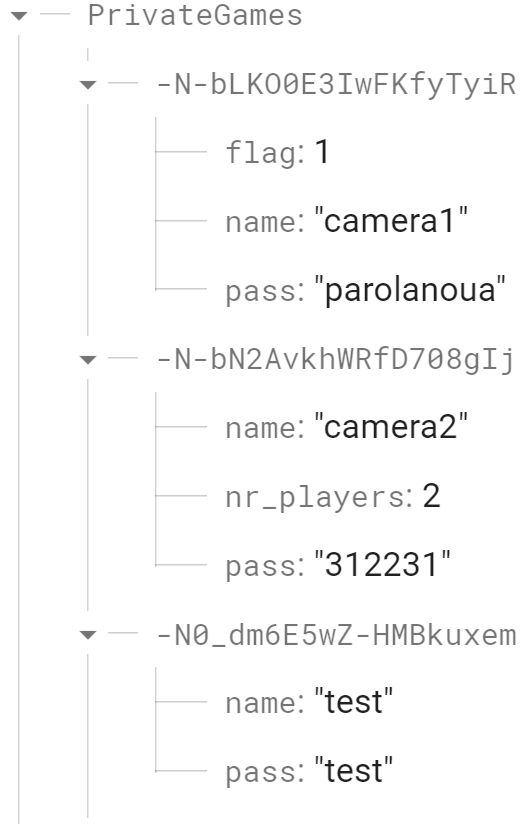
[12] <https://ro.vector.me/browse/180230/database_clip_art>

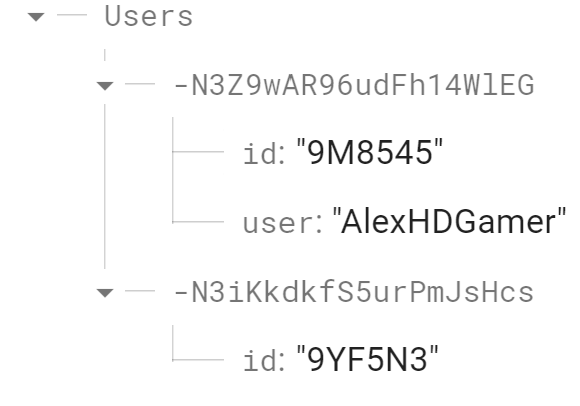
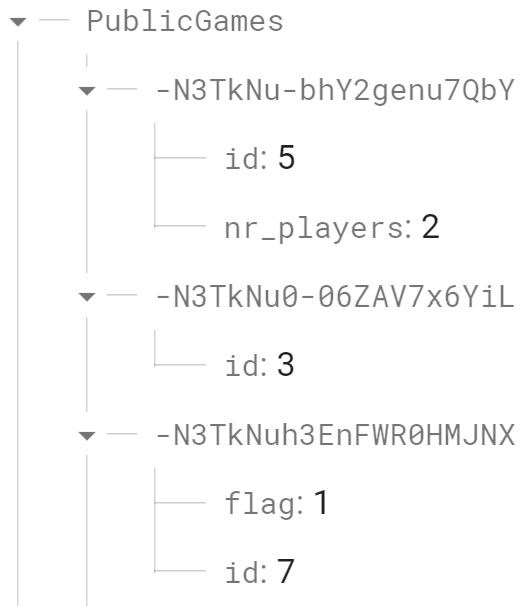
# Anexe

Structuri baze de date

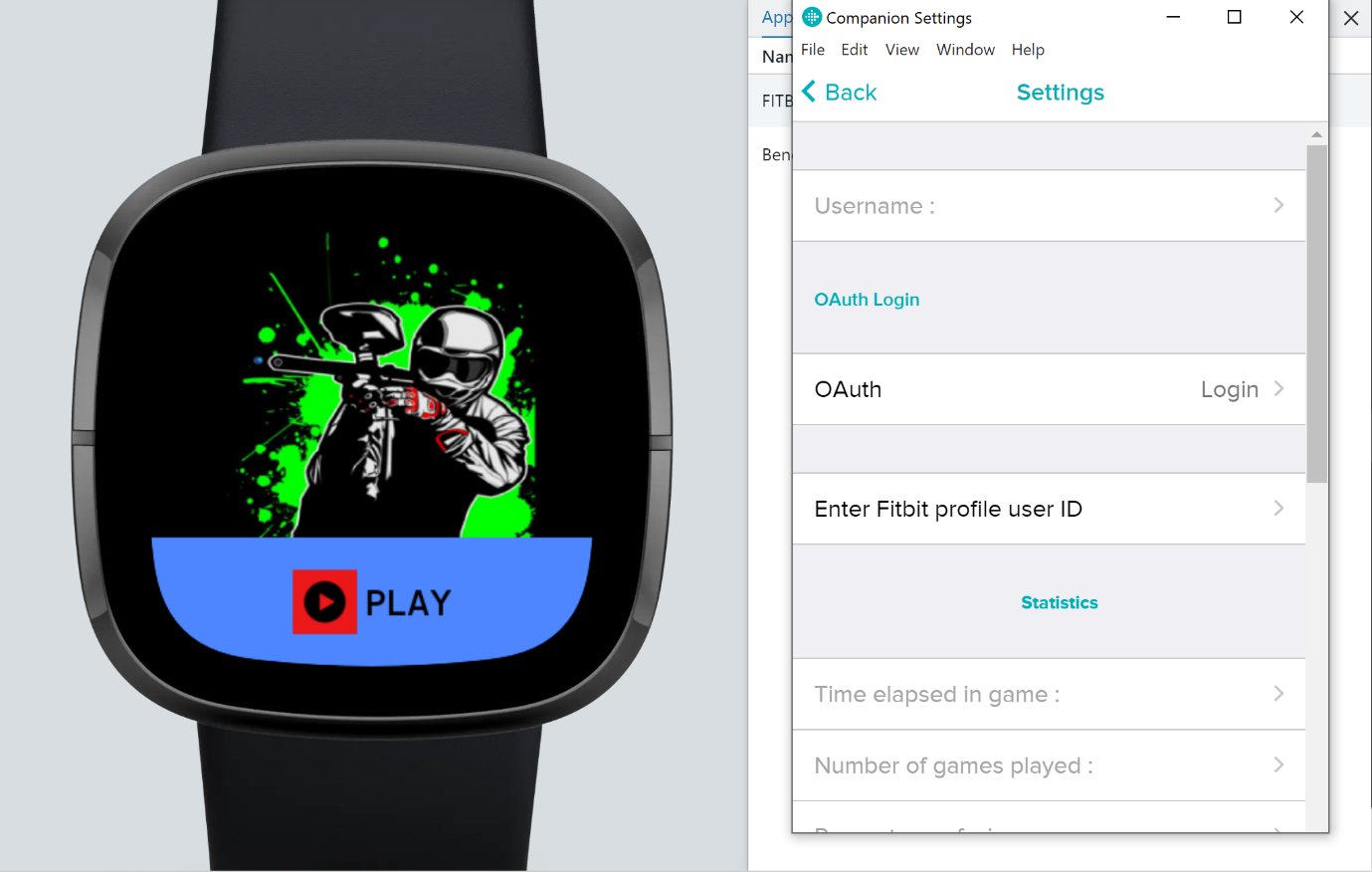
O imagine care conține text

Descriere generată automat

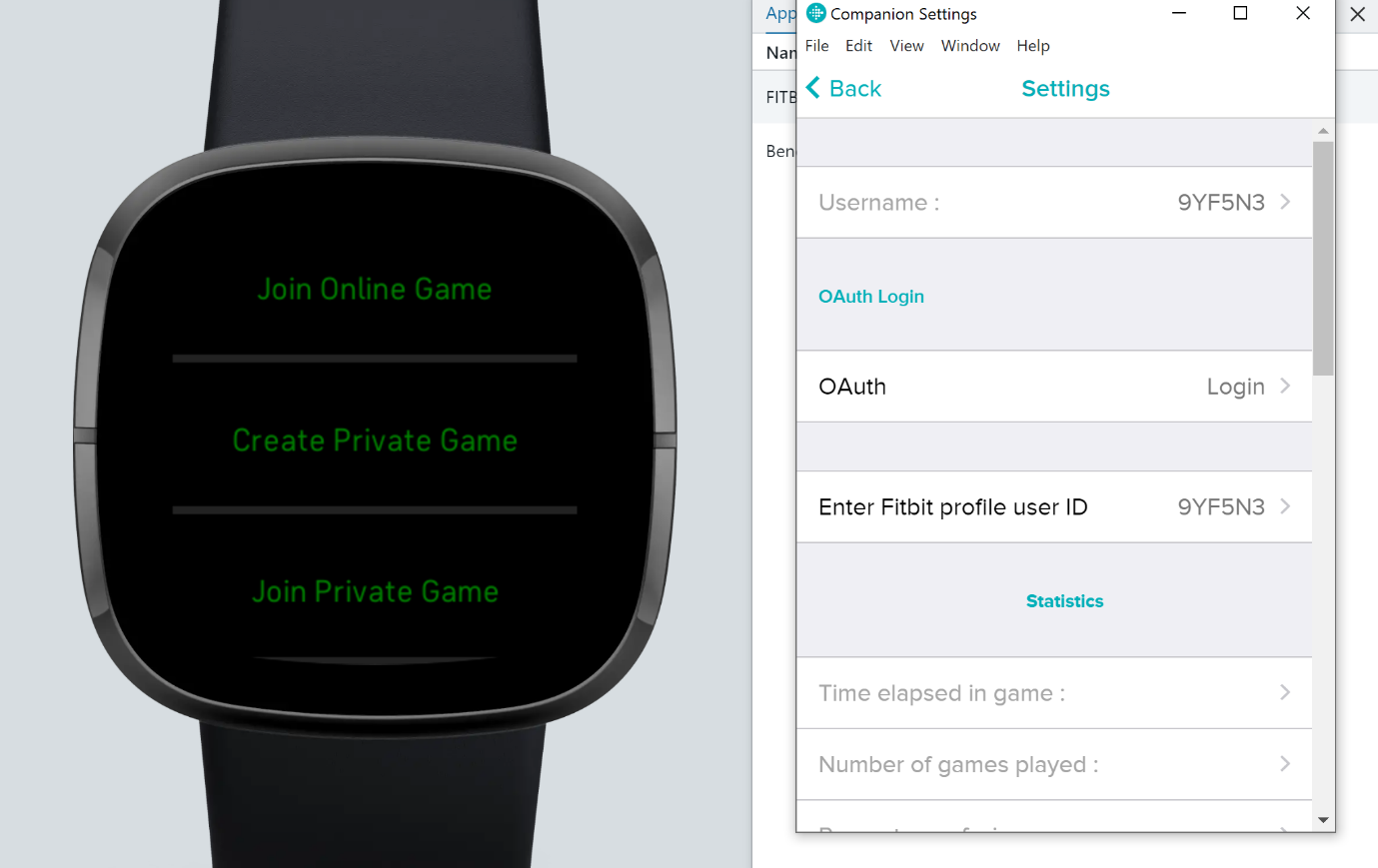




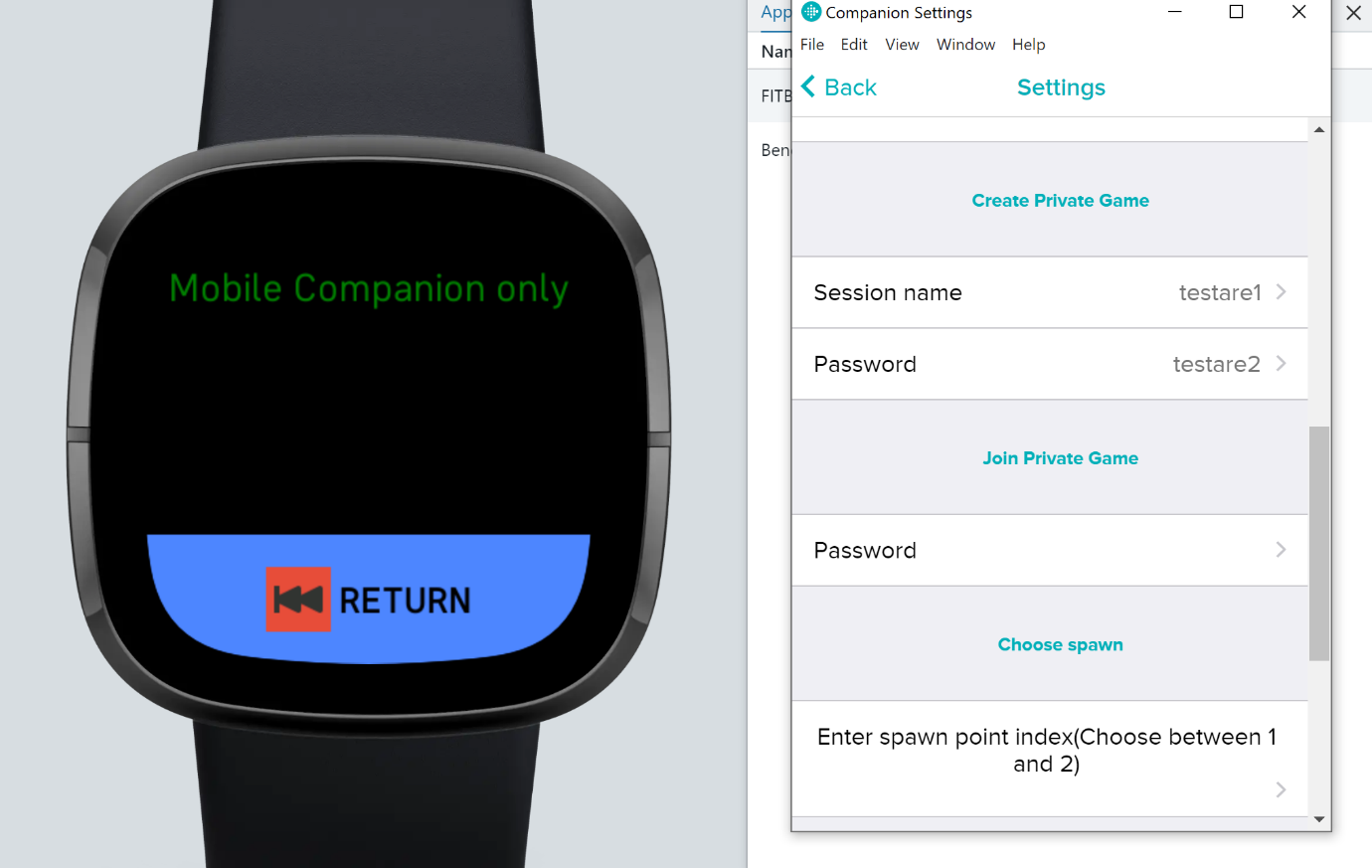
Meniu start



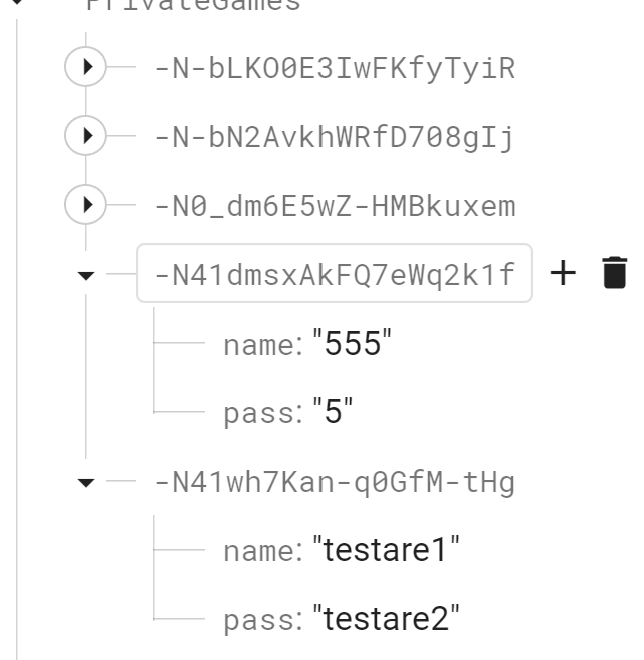
Meniu principal



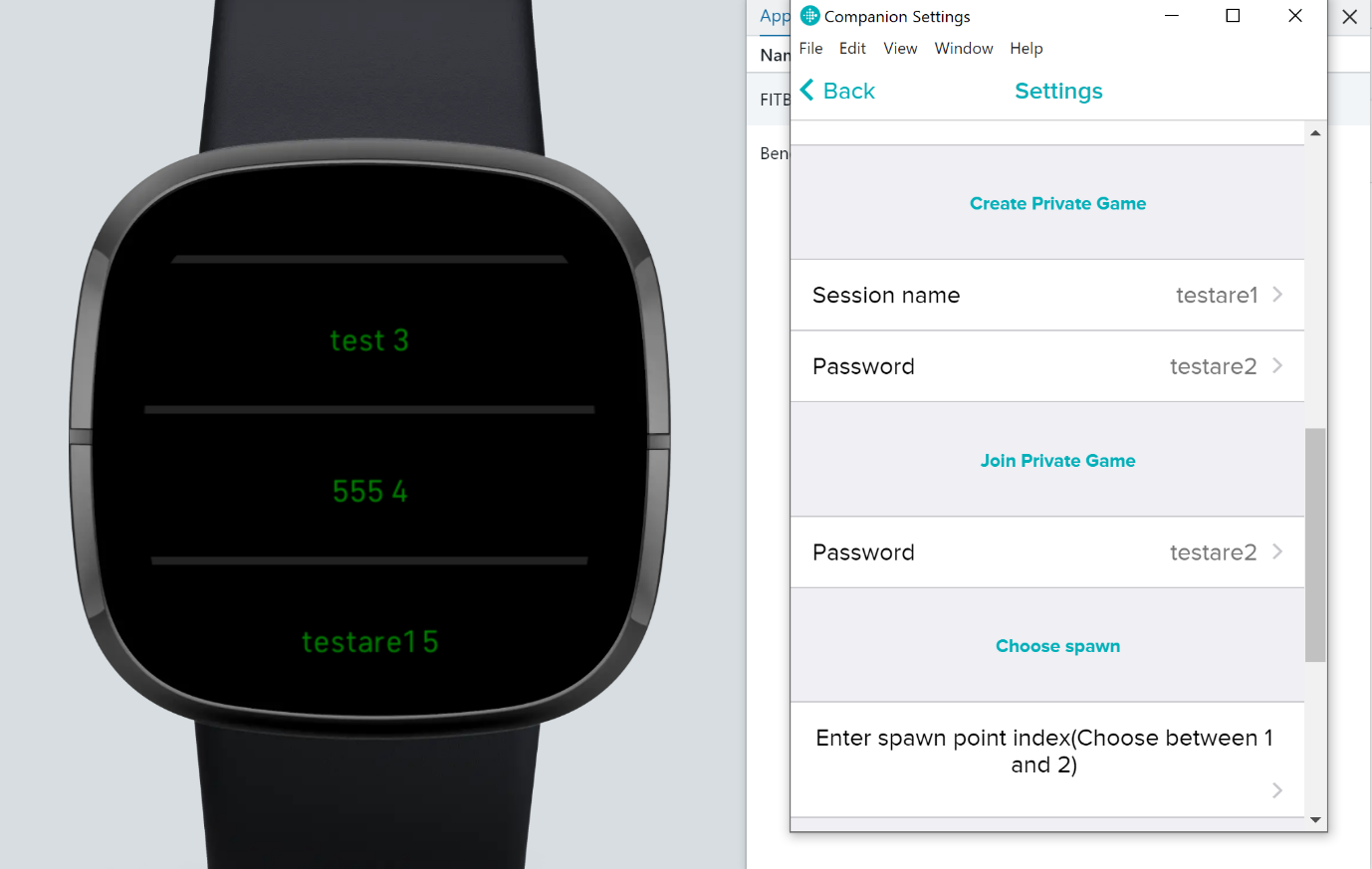
Create Private Game



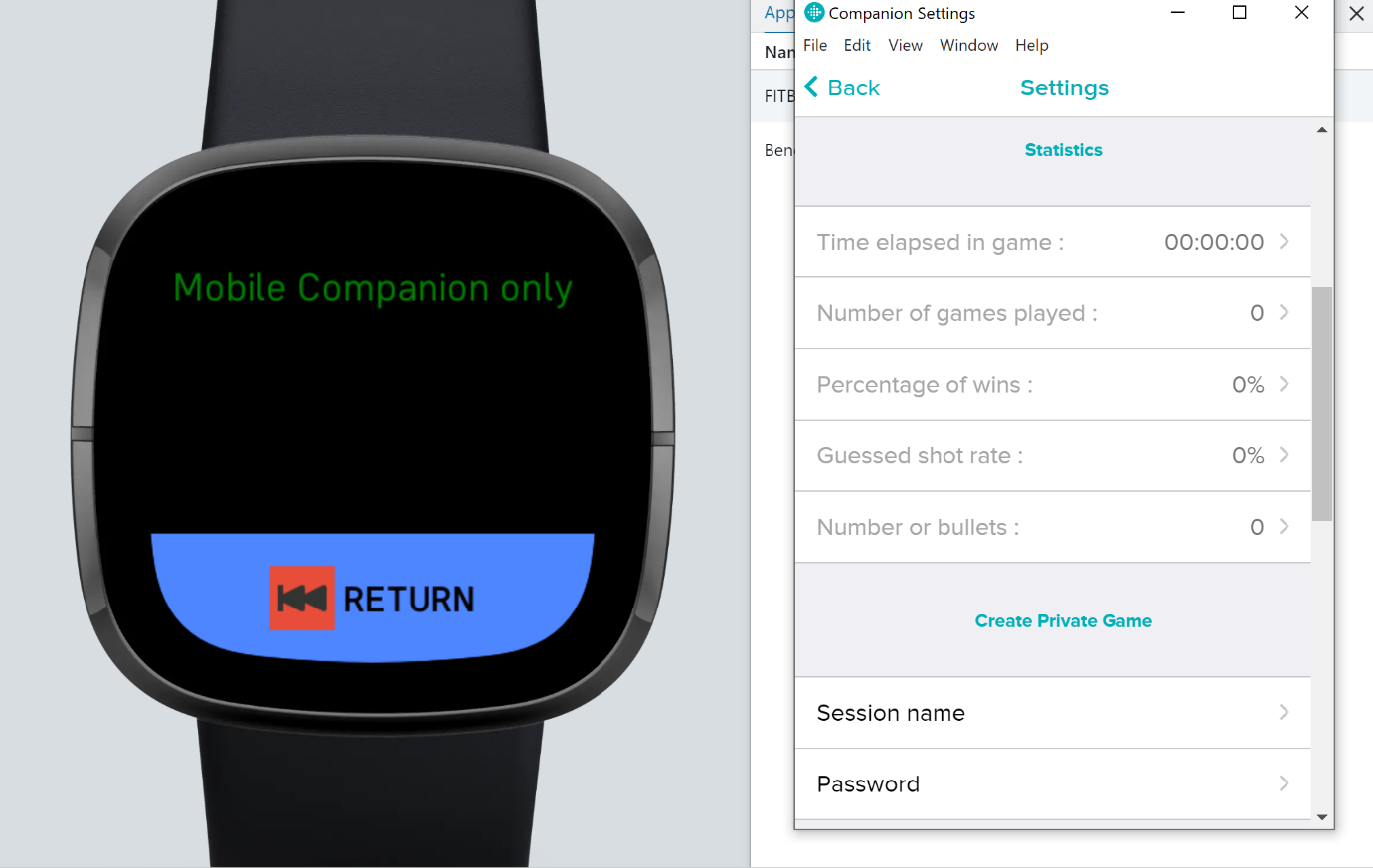
Update în baza de date



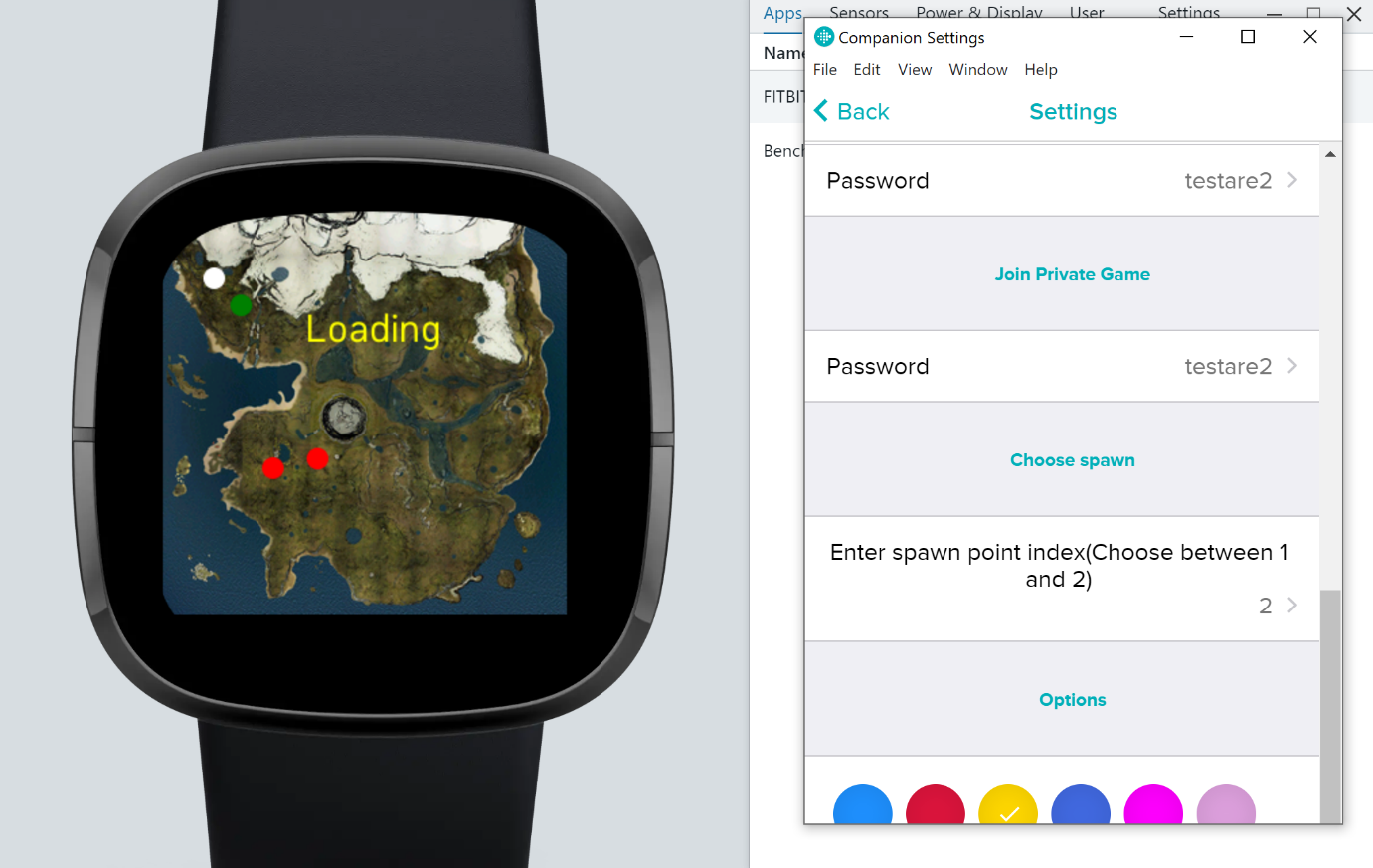
Join Private Game



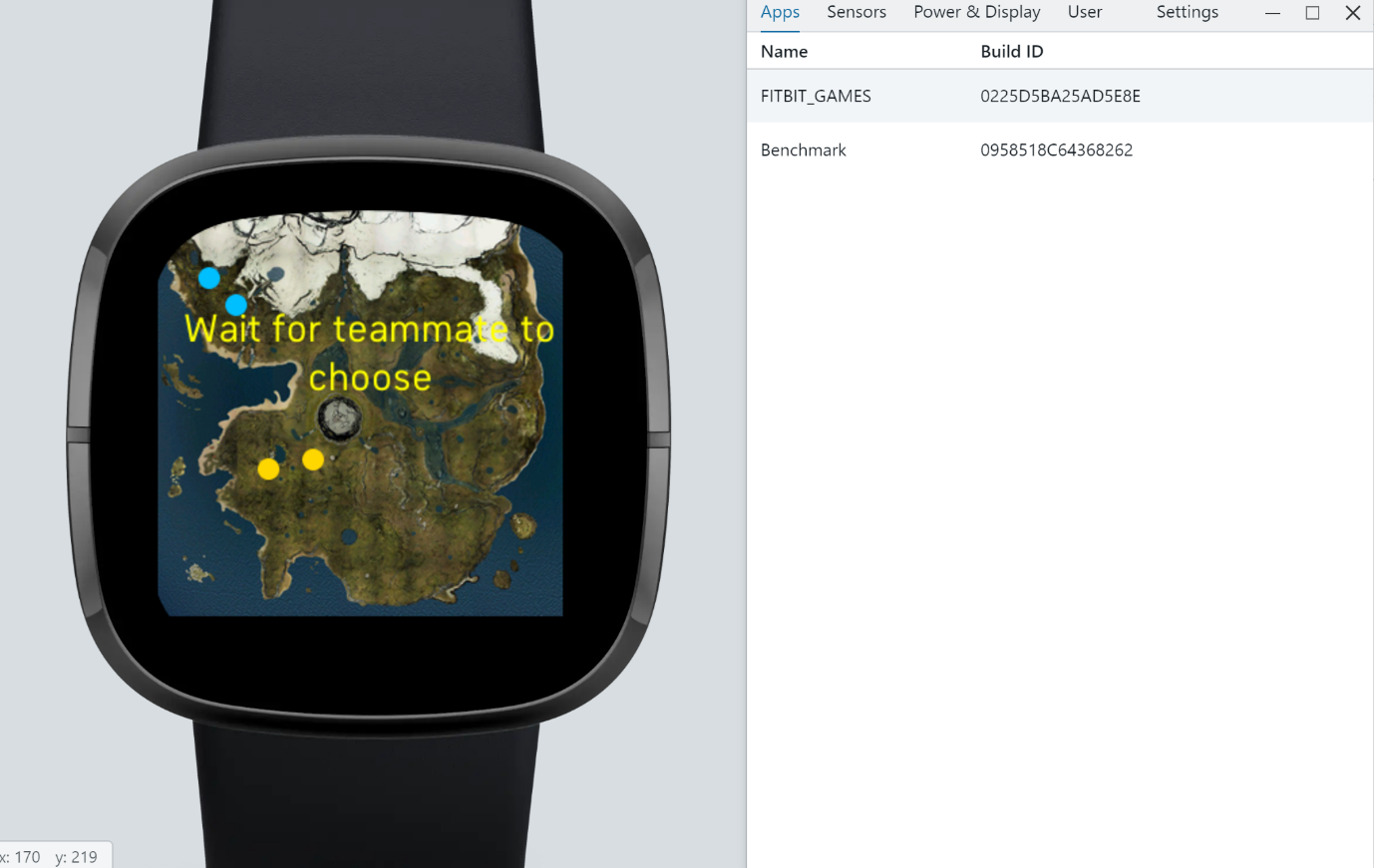
Statistics inițializare



Meniu start joc



Meniu secundar joc



Joc + cum arată o camera în baza de date

O imagine care conține text, ceas, captură de ecran

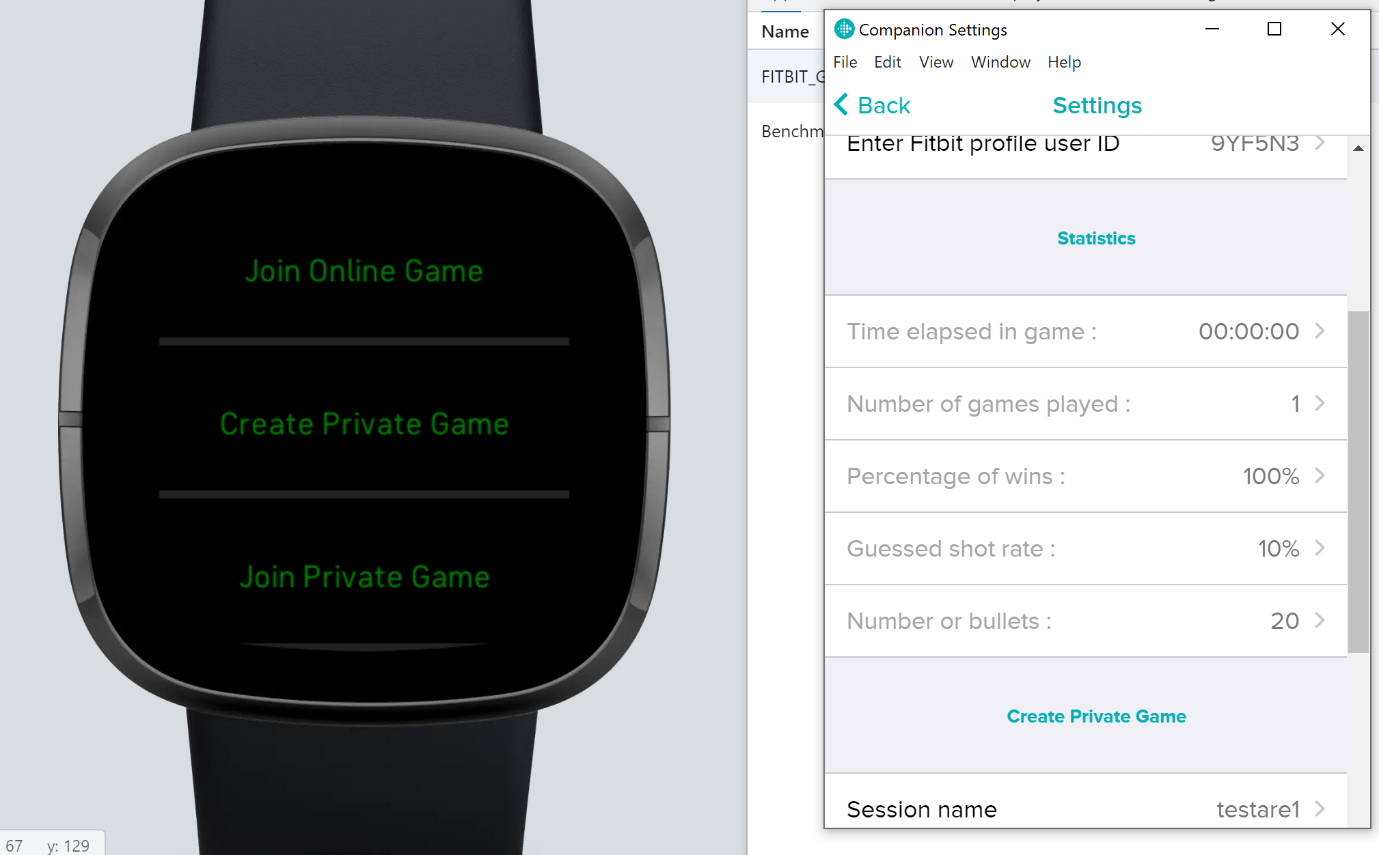
Descriere generată automat

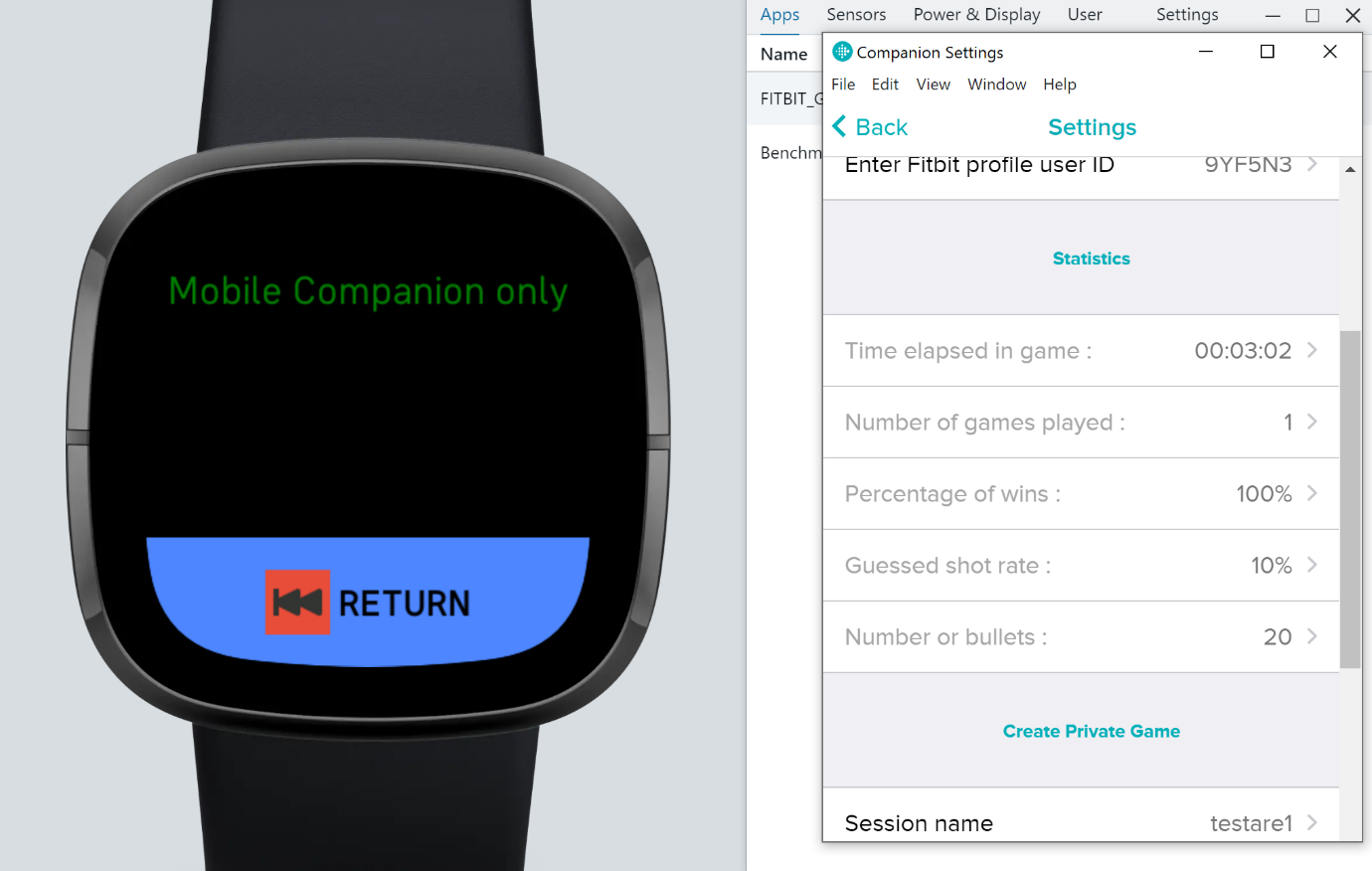
O imagine care conține text

Descriere generată automat



Statistics update





Join Online Game

