Darts - Krikett

Projektfeladat specifikáció

Informatikai Biztonsági és Adatvédelmi Tanácsadó Kft.

1 Tartalomjegyzék

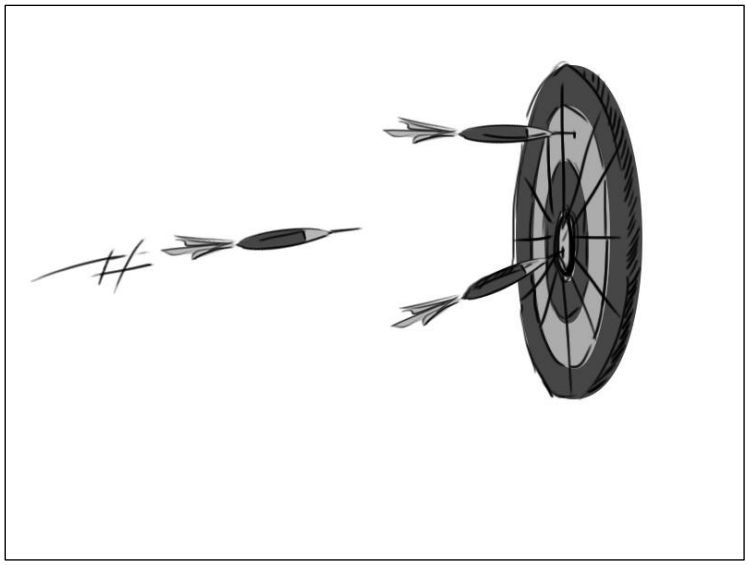
2 Bevezetés

2.1 A feladat címe

Darts - Krikett

2.2 A feladat rövid ismertetése

A projekt célja egy darts krikett játék pontozását segítő mobil alkalmazás elkészítése. A darts nagyon népszerű játék, a táblán többféle játék játszható. Ezek közül a krikett pontozása talán a leginkább bonyolult. Ha a krikettet nem darts gépen játsszuk (vagy steel nyilakkal dobálunk), a pontok számolgatása kényelmetlen feladat. Hogy az adminisztráció ne rontsa a játékélményt, egy olyan grafikus felületű alkalmazást szeretnénk fejleszteni, ami a számolást elvégzi helyettünk.



Amennyiben a csapat mérete, illetve a projekttagok képességei lehetővé teszik, a krikett játék mellé a klasszikus 301, 501 és round-a-clock játékok adminisztrálása is megvalósítható az alkalmazás keretei közt.

3 Elvárások a feladattal kapcsolatban

3.1 Operációs rendszer, környezet

- Minimum Android 11.0 (min. API level 30)

3.2 Felhasználandó programozási nyelv

- Nincs megkötés
- Ajánlott nyelvek, keretrendszerek: Java, Kotlin, Dart; Flutter

3.3 Megoldás formátuma

- Forráskód állományok
- Teljes projekt környezet
- Forráskód dokumentáció
- Üzemeltetői dokumentáció (odt/docx és pdf formátumban)
- Felhasználói dokumentáció
- Erőforrásterv és munkaidő-nyilvántartás

3.4 Szoftverfejlesztés

A feladat egy olyan mobilalkalmazás készítése, mely segít a pontok számolásában a Darts – Krikett játék közben.

A fejlesztés közben be kell tartani a Clean Code alapszabályait.

Az elkészült megoldásnak maradéktalanul meg kell valósítania az 5. fejezetben megfogalmazott követelményeket. Amelyik követelmény nincs pontosan definiálva, ott a megvalósítás során a fejlesztő szabad kezet kap. Fontos viszont, hogy a választott megoldás megfelelő színvonalú legyen mind felhasználói, mind fejlesztői szempontból.

3.5 Modulok

A projekt keretében történő megvalósítás egy lehetséges felbontási lehetősége az alábbi:

- Adatbázis tervezés, kivitelezés, DB interfész megírása
- Felhasználóbarát front end (GUI) tervezése, kivitelezése
    - Játék felület, grafikai elemek
    - Menürendszer
    - Statisztikák
    - Beállítások
- Back end osztályok írása, melyek a játék logikáját valósítják meg
    - Játékmotor megvalósítása
    - Statisztikák összegyűjtése

- Projektvezetéssel kapcsolatos dokumentáció, nyilvántartások vezetése, feladatok összehangolása, felhasználói dokumentáció elkészítése, tesztelés.

Lehetőség szerint a fejlesztői dokumentációkat minden esetben a ténylegesen fejlesztést végző projekttagok készítsék el. A felhasználói dokumentáció külön egységet képezhet, érdemes a teszteléssel összekapcsolni a megfelelő minőség biztosítása érdekében.

4 Játék leírása

4.1 A játék célja

A Krikett célja, hogy a darts tábla számozott szektorait 15-20-ig, valamint a bullt lezárjuk, mielőtt ellenfelünk teszi ezt meg. Egy mező lezárásához háromszor kell eltalálni az adott pontértékű területet. A számozott mezőknél a dupla kettő, a tripla három találatot ér, vagyis pl. egy tripla 20 dobással rögtön lezárul a 20-as szektorunk. Bull esetében a kis bull 1, a nagy bull 2 találatot jelent.

4.2 Pontozás

Ha az egyik játékosnak már le van zárva egy szektora, és nyilával újra eltalálja azt, a dobása a többi játékos számára büntetőpontokat eredményezhet. Amennyiben van olyan ellenfele, akinek még nincs lezárva az adott szektor, akkor az az ellenfél a szektor értékének megfelelő mennyiségű büntetőpontot kap. (pl.: A, B és C játékos játszik, A játékosnak már 2 találata van a 19-es mezőben, B-nek 1, C-nek pedig 3. A játékos következik, dob egy tripla 19-et. Dobásával így lezárja a 19-es szektorát – megvan a 3 találat – és egyben 2*19 pont értékben büntetheti társait. Mivel B játékos 19-es szektora még nyitott, 38 büntetőpontot kap, C viszont egyet sem, mert az érintett szektorát már lezárta.)

4.3 A játék menete

A játékosok a rögzített sorrendben 3-3 nyíllal dobnak a táblára. Amennyiben az aktuális játékos krikett mezőt (mezőket) talált el, a 3 dobás végén minden találatot rögzít az alkalmazás felületén. Ennek során rögtön láthatóvá válik, mely szektorait zárta le (hol nem lehet már büntetőpontot kapnia), illetve dobásának eredményeként mennyire változott az ellenfelek büntetőpontjainak értéke. Amennyiben minden nyila értéktelen mezőt talált, egyszerűen csak átadja a sort a következő résztvevőnek. Ha ez megvan, jöhet a következő játékos.

4.4 A játék vége

Az a játékos nyer, aki hamarabb lezárja az összes krikett szektort (15-20-ig és a bullt), valamint kevesebb büntetőponttal rendelkezik, mint ellenfelei. Ha valaki először zárja le összes szektorát a versenyzők közül, de még nem neki van a legkevesebb büntetőpontja, akkor az ellenfelei le nem zárt szektorain adott büntetőpontokkal játékostársait maga mögé tudja utasítani.

5 Szoftver specifikáció

Az alkalmazás csak *landscape* módon jelenjen meg. A felületet úgy kell elkészíteni, hogy egy szokványos kijelzőméretű készüléken görgetés nélkül kényelmesen jelenjen meg 2-3 játékos esetén. Természetesen, ha több játékos van, akkor azt már nem lehet kényelmesen elhelyezni a kijelzőn, ekkor szükséges a vertikális görgetés. Az alkalmazás tehát megfelelő reszponzivitást nyújtson.

5.1 Megjelenés

- A játék indításkor sorban kérje be a játékosok neveit.
- A játék kinézete táblázatos legyen, melyben bal oldalt egymás alatt jelenjenek meg a játékosok nevei. Az oszlopok legyenek a krikett szektorok értékei az alábbi képhez hasonló módon:

	15	16	17	18	19	20	25
Laci 108	II		I III		III	⊗	
István 91	I	I	III		⊗	II	
Phil Taylor 78			⊗		⊗ II	⊗ II	⊗

5.2 Funkciók

- A játékosok neveinek sorrendjében az egyes játékosok körforgásszerűen, egymás után következve rögzíthetik dobásaik eredményét, amíg a játék véget nem ér.
- Az egyes mezőkben jelenjen meg, hogy melyik játékos hányszor találta el az aktuális szektort.
- A felületen egyértelműen látszódjon, ha egy szektor már le lett zárva.
- Büntetőpontok esetén az ellenfeleknél jelezze a büntetőpontok összesített számát, ez minden adatrögzítés során kerüljön frissítésre.
- Csak az aktuálisan dobó játékoshoz lehessen eredményt rögzíteni.
- Dobás után külön gombbal lehessen a következő játékosra lépni.
- Rossz érték bevitele esetén adjon lehetőséget a rögzített érték visszavonásához.
- Ha a szabályok szerint valaki nyer, az alkalmazás ezt jelezze.
- A játékosok nevei és az eredményük kerüljön eltárolásra.
- Legyen egy toplista az eddigi játékosok eredményeivel.
- A program adjon lehetőséget megjeleníteni az eddig lejátszott játékokat, és azok statisztikáit. Statisztika alatt bármilyen releváns érték átlagolása, minimum és maximum értékek megjelenítése értelmezhető: pl. legkevesebb kör alatt befejezett játék, legtöbb büntetőpont egy körben vagy egy teljes játék alatt, sikeres és sikertelen dobások százalékos aránya stb.

6 Dokumentáció

6.1 Erőforrásterv, munkaidő-nyilvántartás

A specifikáció birtokában a projekt résztvevői készítsenek erőforrástervet. Ez tartalmazza a feladatban részt vevő projekttagokat, akik legyenek hozzárendelve a tervezés során azonosított részfeladatokhoz. Minden részfeladat mellé kerüljön egy munkaidő ráfordítási becslés munkaórában számolva. Ezt a tervet a tényleges fejlesztés előtt le kell adni. A feladat megoldása során az elvégzett munkáról készüljön nyilvántartás részfeladatonként és személyenként a tényleges munkaórák számának megjelölésével. A projekt végén a két dokumentum összehasonlításra, az eltérések elemzésre kerülnek.

6.2 Üzemeltetői dokumentáció

Az üzemeltetői dokumentáció célja, hogy a rendszer üzemeltetőinek támogatást adjon a termék üzemeltetésének elsajátításához. Tartalmazza többek között a rendszer architekturális felépítését (alkalmazás stack elemei és azok közti kapcsolat leírása), az alkalmazás fordításához, fejlesztéséhez és futtatásához szükséges követelményeket, technológiákat, továbbá a konfigurációs állományok leírását (ha vannak).

A dokumentációnak a feladat bonyolultságától függő hosszúságúnak kell lennie, maximális terjedelem nincs meghatározva.

6.3 Forráskód dokumentáció

A fontosabb függvények és osztályok előtt szerepelnie kell megjegyzéseknek, melyeknek tartalmazniuk kell az azt követő metódus rövid szöveges – akár magyar nyelvű – leírását. A forráskód dokumentációt a munka során folyamatosan kell készíteni.

6.4 Felhasználói dokumentáció

Az alkalmazás használatának részletes bemutatása képernyőképekkel, funkciók pontos leírásával.

7 A projekt értékelése

7.1 A feladat értékelésének felhasználó oldali szempontjai

A működő alkalmazás tesztelése alapján az alábbiak a legfontosabb jellemzők:

- Kiírást teljes egészében lefedő funkcionalitás
- Kényelmes használat
- Igényes felhasználói felület
- Stabil működés
- Igényes felhasználói dokumentáció

7.2 A feladat értékelésének technikai szempontjai

Informatikai szakmai szempontból a megoldás értékelésének alapja:

- Kódkép, a kód tisztasága, kommentelés minősége
- Kódolási konvenciók betartása (Clean Code)
- Dokumentációk színvonala
- Dokumentált tesztelés
- Erőforrás felhasználásának pontos nyilvántartása

7.3 Projekt megvalósításának piaci jellegű értékelése

A projekt lezárultával összehasonlításra kerül a kezdeti erőforrásterv, valamint a megvalósítás során dokumentált munka. Ezen dokumentumok elemzéséből levezetésre kerülnek azok a problémák, melyek a piaci környezetben jellemzően megjelennek. Végigtekintjük ezen problémák okait, következményeit, lehetséges elkerülésüknek vagy hatásuk mérséklésének módjait. A jellemző hibák ebből a megközelítésből:

- Határidő csúszása
- Nem megfelelő minőség
- Hiányos, vagy elmaradó tesztelés
- Használhatatlan, pontatlan dokumentáció
- Pontatlan erőforrás becslés
- Aránytalanul magas önköltség
- Az elkészült termék továbbfejlesztésének, karbantartásának nehézségei

A fentiek értékelésén túl fejlesztői szemszögből elemezzük a megvalósítás tapasztalatait, a lehetséges továbbfejlesztés, átalakítás, támogatás kérdéseit és piaci lehetőségeit.

8 Projekt adatlap

Projekt neve: Darts - Krikett

Feladat rövid ismertetése: Darts krikett játék pontozását segítő mobilalkalmazás

Specifikációt összeállította: Szabó Gábor Ferenc, Apáti László, Varga Bence