**Baranya Megyei Szakképzési Centrum**

**Baranya Megyei SzC Radnóti Miklós Közgazdasági Technikum**

**Vizsgaremek**

Készítették: Pópé Dániel

és Kapus Benjámin

**Pécs**

**2023**

**Baranya Megyei Szakképzési Centrum**

**Baranya Megyei SzC Radnóti Miklós Közgazdasági Technikum**

Szakma megnevezése: Szoftverfejlesztő és –tesztelő

A szakma azonosító száma: 5 0613 12 03

**Vizsgaremek**

Szakikereső

Készítették: Pópé Dániel

és Kapus Benjámin

**Pécs**

**2023**

Tartalomjegyzék

[I. Bevezető, a feladat rövid ismertetése 4](#_Toc132298484)

[II. A felhasználói dokumentáció 4](#_Toc132298485)

[1. A program általános specifikációja 4](#_Toc132298486)

[2. Rendszerkövetelmények 4](#_Toc132298487)

[• Hardver követelmények 4](#_Toc132298488)

[• Szoftver követelmények 4](#_Toc132298489)

[3. A program telepítése 5](#_Toc132298490)

[4. A program használatának a részletes leírása 5](#_Toc132298491)

[III. A fejlesztői dokumentáció 5](#_Toc132298492)

[1. Témaválasztás indoklása 5](#_Toc132298493)

[2. Az alkalmazott fejlesztői eszközök 6](#_Toc132298494)

[3. Tervezési módszer 6](#_Toc132298495)

[4. Adatmodell leírása 6](#_Toc132298496)

[5. Részletes feladatspecifikáció, algoritmusok 6](#_Toc132298497)

[6. Forráskód 7](#_Toc132298498)

[7. Tesztelési dokumentáció 7](#_Toc132298499)

[8. Továbbfejlesztési lehetőségek 7](#_Toc132298500)

[IV. Összegzés 7](#_Toc132298501)

# I. Bevezető, a feladat rövid ismertetése

A vizsgaremekhez több projektötletünk volt, viszont a túlzott bonyolultság és a követelmények alapján szépen kiszűrtük az ötleteket és végül erre a témára esett a választásunk.

A mai világban az emberek törekednek a technológiák fejlesztésére, a dolgok praktikusabbá tételére. Ennek egyik hátránya, hogy minden egyre bonyolultabb és összetettebb lesz, ezért mindenhez szükség van szaktudásra vagy speciális eszközökre. A dolgok elhasználódnak, elavulnak, vagy tönkre mennek.

Bizonyára mindannyian, vagy legalábbis a családban, ismerettségi körben előfordult, hogy felfogadtunk valamilyen szakembert. Ezt sok keresés, kevés választási lehetőség és akár több hét, hónap várakozás előzte meg. Emellé természetesen hozzátartozik az is, hogy a szakember milyen munkát végzett. Sajnos gyakran hallunk, megbízhatatlan, munkáját rosszul elvégző emberekről, akik nagyobb kárt okoztak, mint amennyi hasznot. Ezekre a problémákra szeretnénk megoldásként ezt a projektet.

A fejlesztés során rengeteg új ismeretet szereztünk, mint például a kód lefutásakor keletkezett hibák megoldásáról. Emellett több olyan része is volt a projektnek, amivel az iskolában nem találkoztunk és ezért saját magunknak kellett rájönni, megkeresni a megoldást. Összességében egy jó „tanulópénz” volt ez a projekt.

A projekttel kapcsolatban felmerült az, hogy publikus és elérhető legyen mindenki számára az interneten, viszont a tapasztalatlanság és a tőke hiánya miatt arra a döntésre jutottunk, hogy erre nem kerül sor. A projekt viszont teljes mértékben tovább fejleszthető, ötletekre a későbbiekben kitérek.

A feladatok elosztását alapvetően két részre osztottuk: frontend és backend. Az ötletelési, tervezési szakasznál jobbnak találtuk, hogy teljesen együtt dolgozzuk ki az adott feladatot. Az adatbázis-táblák is így jöttek létre. Ezt követően a kinézettel és az azokhoz tartozó, böngészőben megvalósítható funkciókkal Pópé Dániel foglalkozott. Az adatbázis eljárások készítésével és az adatbázis kapcsolat létrehozásával Kapus Benjámin dolgozott. Természetesen végig kommunikáltunk és ahol tudtunk besegítettünk a másiknak.

A projektmunka során több projektszervezési programot is használtunk. Például Trello-t az ötletek rendszerezéséhez, GitHub-ot a projekt tárolására és verziókezelésére.

# II. A felhasználói dokumentáció

A felhasználói dokumentáció célja, hogy segítséget adjon a leendő felhasználónak a program telepítésével és használatával kapcsolatos minden probléma megoldásában, segítse a program kezelésének az elsajátítását.

A felhasználói dokumentációnak a következő részeket kell tartalmaznia:

## 1. A program általános specifikációja

Ez a rész a program fontosabb jellemzőit és funkcióit tartalmazza. A cél, hogy a leendő felhasználó ezt a fejezetet elolvasva el tudja dönteni, hogy a program megfelelő-e a számára.

Ajánlott terjedelem: 1-2 oldal.

## 2. Rendszerkövetelmények

### • Hardver követelmények

Ebben a részben kell leírni a minimális és ajánlott hardver konfigurációt, amely a program futtatásához szükséges. Pontos paramétereket kell megadni, még akkor is, ha a program amúgy minden gépen lefut.

### • Szoftver követelmények

Le kell írni, hogy mely operációs rendszere(ke)n fut a program, és milyen egyéb szoftver komponensek szükségesek a működéshez (pl. .NET, DirectX, esetleg adatbázis-szerver és adatbázis-állományok, stb.) A beadott CD-n ezeknek is ott kell lenniük.

Ajánlott terjedelem: ½ -1 oldal, felsorolásszerűen leírva

## 3. A program telepítése

• Képekkel illusztrált, részletes leírás a program telepítésének a menetéről.

• A leírás alapján a felhasználónak hiba nélkül telepíteni kell tudni a programot. A leírásnak ki kell térnie a telepítés során kiválasztható opciókra is.

• Ha esetleg nincs telepítőprogram, akkor kellő részletességgel le kell írni, hogy mely fájlokat, pontosan hova kell felmásolni, és hogy lehet a programot futtatni.

Ajánlott terjedelem: 2 -4 oldal, ábrákkal együtt.

## 4. A program használatának a részletes leírása

Mindenre kiterjedő, részletes leírás a program használatáról.

Alapszabályok:

• Amit leprogramozott, azt a dokumentációban is írja le, ne legyenek eltitkolt funkciók.

• Minden pontosan, „szájbarágósan” legyen leírva. A dokumentáció alapján a teljesen kezdő, vagy laikus felhasználóknak is használniuk kell tudni a programot.

• A stílus legyen pontos és közérthető, vegye figyelembe, hogy a felhasználói dokumentáció nem szakembereknek készül.

• Ugyanakkor kerülje a laza stílust: rövidítések, smilie-k, szleng kizárva.

• Alkalmazzon ábrákat, screenshot-okat , de a ne legyen túlzott a képek aránya a szöveghez képest. Kb. oldalanként egy ábra megfelelő (nem nagy).

Ajánlott terjedelem: 12-17 oldal, ábrákkal együtt.

# III. Fejlesztői dokumentáció

A kód írásakor törekedtünk a „clean code” elveinek megfelelni. Ennek fényében a lehető legegyszerűbb, legrövidebb és legjobb megoldásokat alkalmaztuk. A sorok, függvények köré megjegyzéseket is tettünk a még gyorsabb megértés érdekében. Az eredményen látszik, hogy könnyen átlátható és megérthető kódot sikerült készítenünk.

## 1. Témaválasztás indoklása

A téma kiválasztásának fő szempontjai természetesen a vizsgaremek elvárásai voltak, viszont fontosnak tartottuk, hogy hozzánk közel álló, ismert problémára, feladatra találjunk megoldást. Ezért is sikerült egy életszerű problémára megoldást nyújtó oldalt létrehozni.

Adatbázist is használ, mivel a felhasználók, a hirdetések és sok más adatot el kell tárolnunk. Ezekhez az adatokhoz rengeteg rész tartozik: A regisztrációtól kezdve a hirdetések lekérésén át az értékelések írásáig bármi. Vannak alap és különleges funkciók. Tehát sok lehetőség és elképzelés volt. Ezekre a függvényekre könnyen rálehetett építeni a RESTful komponenseket is.

Ezen kívül volt rendelkezésre álló megjeleníthető tartalom is, mint például a hirdetések, kedvencek, profil adatok. Ezekhez az elemekhez könnyű volt a megjelenítést kialakítani, ami különböző méretű eszközökön is jól látható és áttekinthető.

## 2. Az alkalmazott fejlesztői eszközök

* Programnyelvek
  + Adatbázis: MySql
  + Frontend: Angular
  + Backend: Java
* Fejlesztői környezet
  + Adatbázis: Xampp (PHPMyAdmin)
  + Frontend: Visual Studio Code, PostMan, Paint
  + Backend: Netbeans, PostMan, WildFly szerver
  + Mindegyik: Git, GitHub Desktop

## 3. Tervezési módszer

• Alkalmazás tervezése,

• tervezési módszertan ismertetése,

• vizuális modellezés

• OOP megvalósulásának leírása, részletezése

Ajánlott terjedelem: a feladat jellegétől függően 2-5 oldal.

## 4. Adatmodell leírása

A feladatban alkalmazott adatbázis vagy adatszerkezet részletes leírása

• Adatbázis esetén az adattáblák leírása, a közöttük lévő kapcsolatok megadása. Az adatbázis adatbázismodell-diagramja.

• Normalizálás folyamatának leírása

• OOP jellegű megvalósítás: UML osztálydiagram

• Biztonsági kérdések, titkosítás, GDPR

• Ajánlott terjedelem: a feladat jellegétől függően 2-5 oldal.

## 5. Részletes feladatspecifikáció, algoritmusok

• A program lényeges függvényeinek, az osztályok metódusainak a specifikációja (mit valósít meg az adott függvény, illetve metódus, milyen paraméterei vannak, mi a visszatérési érték)

• Az algoritmizálható részek leírása valamilyen algoritmus-leíró eszközzel (struktogram, pszeudo-kód, esetleg UML aktivitás-diagram)

Ajánlott terjedelem: a feladat jellegétől függően 2-5 oldal.

## 6. Forráskód

• A teljes forráskódot a nyomtatott dokumentációba nem kell beletenni!

• Lehet viszont a nyomtatott dokumentációban a fontosabb kódrészeket magyarázattal szerepeltetni

## 7. Tesztelési dokumentáció

• Részletes tesztelési dokumentáció készítése

• A tesztelés során kiderült hibák felsorolása

A tesztelési dokumentációból derüljön ki, hogy ismeri a különböző tesztelési módszereket (pl. fekete doboz, fehér doboz módszer), tesztelési szinteket, stb.

Ajánlott terjedelem: a feladat jellegétől függően 5-7 oldal.

## 8. Továbbfejlesztési lehetőségek

• Olyan ötletek, amelyeket meg akartak valósítani, de nem sikerült, vagy nem fért bele az időbe

• Olyan ötletek, amelyeket még érdemes a jövőben megvalósítani

Ajánlott terjedelem: 1-2 oldal

# IV. Összegzés

Röviden leírhatja, hogy a vizsgaremek milyen területeken segített fejlődni, mind szakmailag és egyéb területen. Milyen kihívásokkal kellett megküzdened az elkészítés során, a témával kapcsolatban, milyen jövőbeli célok fogalmazódtak meg.

8. Irodalomjegyzék, forrásmegjelölés

• Minden olyan forrás pontos megadása, amelyet a vizsgaremekben felhasznált.

• A forrás lehet pl.

• Könyv. Meg kell adnia a következőket: szerző(k), cím, kiadó, kiadás éve

• Weboldal. Meg kell adni a linket, az oldal címét

• Elektronikus dokumentum. Meg kell adni a szerzőt, a letöltés helyét, idejét

Ajánlott terjedelem: ½ -1 oldal.