



ÓBUDAI EGYETEM
OBUDA UNIVERSITY

ORSZÁGOS TUDOMÁNYOS DIÁKKÖRI DOLGOZAT

MESTERSÉGES INTELLIGENCIÁVAL TÁMOGATOTT VIZSGAADMINISZTRÁCIÓS RENDSZER AUTOMATIKUS TESZTÉRTÉKELÉSEL

Szerző: Moldovan Evelin Bernadett

Óbudai Egyetem, mérnökinformatikus BSc.
szak, V. évfolyam, 2024.11.

Témavezető: Kovács András

Budapest, 2024

TARTALOMJEGYZÉK

1	BEVEZETÉS	4
1.1	Téma bemutatása	4
1.2	Szkennelő alkalmazás	4
1.3	Javító alkalmazás	5
2	HASONLÓ RENDSZEREK ELEMZÉSE	6
2.1	Gradescope.	6
2.1.1	Mi az a Gradescope?	6
2.1.2	Hogyan működik?	7
2.1.3	Funkciók	8
2.2	ZipGrade	9
2.2.1	Mi az a ZipGrade?	9
2.2.2	Hogyan működik?	10
2.2.3	Funkciók	10
2.3	Konklúzió	11
3	FELHASZNÁLHATÓ KÉPFELDOLGOZÓ ELJÁRÁSOK	12
3.1	Képek előfeldolgozása	13
3.1.1	Kép elforgatása	13
3.1.2	Releváns részek kivágása	18
3.1.3	Értékelés.	19
3.2	Karakterfelismerés.	20
3.2.1	Tesseract OCR	20
3.2.2	Yolo algoritmus	20

3.2.3	Transformer-alapú Optikai Karakterfelismerés	22
3.3	Konklúzió	23
4	RENDSZERTERV	25
4.1	Szkennelő alkalmazás: Neptun kódok és szkennelt vizsgafeladatok kezelése	25
4.1.1	Cél	25
4.1.2	Funkcionális követelmények	25
4.2	Szerveroldali és -javító alkalmazás	25
4.2.1	Cél	27
4.2.2	Funkcionális követelmények	27
4.3	Részletes rendszerterv	27
4.3.1	Angular kliens és szerveroldali ASP.NET alkalmazás kommunikációja	28
4.3.2	Összegzés	30
5	MEGVALÓSÍTÁS	31
5.1	WPF alkalmazás.	31
5.1.1	Adatok feldolgozása	32
5.1.2	Adatok továbbítása a szerver felé	33
5.2	Szerveroldali és Javító alkalmazás.	34
5.2.1	Szerveroldali feldolgozás	34
5.2.2	Angular kliens	34
6	TESZTELÉS ÉS ELÉRT EREDMÉNYEK	37
6.1	Tesztelési környezet	37
6.2	Adatok elemzése	37
6.2.1	Pontosság és hibaarány	38
6.2.2	Sebesség és feldolgozási idő	39

6.2.3	Korlátok és relevancia	39
6.3	Eredmények összegzése.	40
7	TOVÁBBFEJLESZTÉSI LEHETŐSÉGEK	43
8	IRODALOMJEGYZÉK	46
9	ÁBRAJEGYZÉK.	48
10	RÖVIDÍTÉSEK	49

1 BEVEZETÉS

1.1 Téma bemutatása

Az oktatási intézményekben használt papíralapú vizsgák adminisztrációja jelentős kihívásokkal jár mind az időigényesség, mind a komplexitás tekintetében. Ezen folyamatok optimalizálása, különösen a vizsgák javításának és tárolásának digitalizálása, nagy mértékben hozzájárulhat az oktatási intézmények hatékonyságához. A papíralapú vizsgák lebonyolítása során felmerülő problémák, mint például a dokumentumok fizikai kezelése, archiválása, valamint a javítási folyamatok hosszadalmas volta, komoly terhet jelentenek az oktatók és az adminisztrátorok számára. Emellett a papíralapú dokumentumok hosszú távú megőrzése további nehézségeket vet fel.

A kutatás célja egy olyan digitális vizsgaadminisztrációs rendszer tervezése és fejlesztése, amely jelentősen leegyszerűsíti és felgyorsítja a papíralapú vizsgák javítási és archiválási folyamatait. A tervezett rendszer olyan technológiai megoldásokat integrál, amelyek lehetővé teszik a vizsgalapok digitalizálását, valamint a digitalizált dokumentumok hatékony - és tesztfeladat esetén automatizált - javítását. Az oktatási intézményekben rendelkezésre álló szkennelési infrastruktúra támogatja a papíralapú vizsgalapok szkennelését, és a felhasználók számára X darab JPG formátumú fájl létrehozását.

Összességében a tervezett digitális vizsgaadminisztrációs rendszer kifejlesztése jelentős hatással lehet az oktatási intézmények működésére, különösen a vizsgák lebonyolításának és értékelésének terén. A rendszer hozzájárulna a tanárok munkaterhelésének csökkentéséhez, valamint az oktatás minőségének javításához. A munkafolyamatok gyorsítása lehetővé teszi az oktatók számára a gyors és hatékony visszajelzés nyújtását a hallgatók felé, ezzel segítve a tanulási folyamatukat.

1.2 Szkennelő alkalmazás

A szkennelő alkalmazás fő funkciói a beszkenelt vizsgalapokon található feladatok körbevágása feladattípus alapján, az azonosítók és Neptun kódok minél pontosabb felismerése, a tesztfeladatok automatikus javítása, valamint ezen adatok felküldése a szerverre API-n keresztül.

1.3 Javító alkalmazás

A második szoftver egy olyan alkalmazás, amelybe az oktatók az O365-ös címükkel léphetnek be és megfelelő jogosultság megléte mellett elérhetik a javítandó vizsgáikat. Ez az integráció lehetővé teszi a felhasználók számára, hogy könnyen és biztonságosan hozzáférjenek az alkalmazáshoz az általuk használt Office 365 bejelentkezési adataikkal.

2 HASONLÓ RENDSZEREK ELEMZÉSE

2.1 Gradescope

Oktatóként az egyik legfontosabb, de gyakran legtöbb időt igénylő szerep az osztályozás: megoldott dolgozatok javítása és hatékony visszajelzés nyújtása. A hatékony visszacsatolás jellemzője, hogy gyors, megbízható, magyarázó és célja a hiányosságok azonosítása és feloldása az aktuális és a kívánt teljesítmény között.

Ezen jellemzők alapján a visszajelzésnek gyorsnak kell lennie, hogy azonnal reagálhassunk a diákok munkájára és lehetőség szerint azonnali segítséget nyújthassunk, meg kell bízunk abban, hogy az értékelésünk igazságos és méltányos, biztosítva ezzel a diákok számára a teljesítményük objektív mérését, a visszajelzésnek magyarázónak kell lennie, hogy a diákok megértsék, hol és hogyan javíthatnak, illetve azonosítania kell a hiányosságokat, amelyek megakadályozzák a diákokat abban, hogy elérjék a kívánt teljesítményt. Atwood és Singh, labelyearlabelyeardivisionlabelmonthlabelday

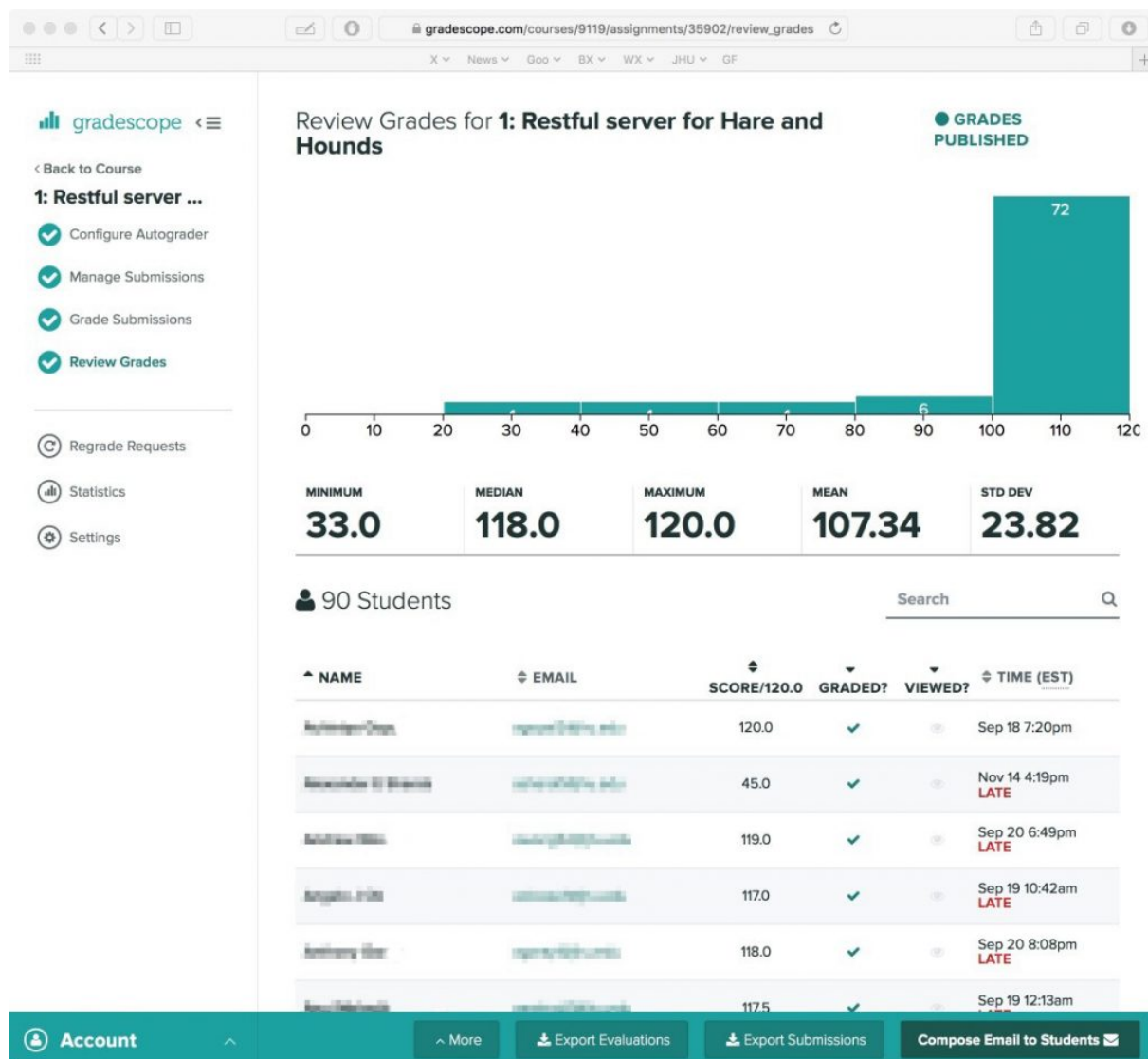
2.1.1 Mi az a Gradescope?

A Gradescope platformot a Edwards, labelyearlabelyeardivisionlabelmonthlabelday forrásmunkában leírtak alapján ismertetem. A Gradescope egy olyan digitális eszköz és online platform, amelyet az oktatók és diákok egyaránt használhatnak a munka beküldésére, osztályozására és értékelésére. Ez az innovatív megoldás lehetővé teszi az oktatók számára, hogy egyszerűen hozzáférjenek és értékeljék a diákok által benyújtott feladatokat és vizsgákat. Az alkalmazás és a felhőalapú platform révén bárhol és bármikor elérhető, lehetővé téve a hozzáférést és a munka folyamatos nyomon követését.

A Gradescope nem csupán egy digitális eszköz, hanem lehetőséget biztosít a papíralapú munkák digitalizálására is. A rendszerbe való beszkenelés révén a papíralapú feladatok könnyedén importálhatók és kezelhetők, így biztosítva a hatékony visszajelzést és értékelést.

A platform számos beküldési típust támogat, ideértve a hagyományos feladatokat, vizsgákat és akár kódolási feladatokat is. Az értékelés gyors és hatékony, miközben lehetőséget nyújt a részletes visszajelzésre és megjegyzések írására is, amely segíti a diákok fejlődését. A rendszer rubrikákon és kérdéseken alapuló elemzést kínál, amely segítségével az oktatók pontos képet

kaphatnak az egyéni teljesítményről és a csoport általános eredményeiről, ahogyan azt a 2.1. ábra is bemutatja.



2.1. ábra: Gradescope alkalmazás Atwood és Singh, *labelyearlabelyeardivisionlabelmonthlabelday*

2.1.2 Hogyan működik?

Az alkalmazás ajánl néhány ingyenes konvertáló eszközt, amelyek segítségével a diákok könnyen átalakíthatják kézzel írt munkájukat PDF formátumba.

Miután a diákok feltöltötték munkájukat, az alkalmazás automatikusan azonosítja a kézzel írt neveket, és meghatározza, hol kezdődik és végződik a munka. Ezután a tanároknak lehetőségük van a beküldött anyagok anonimizálására, hogy valóban elfogulatlan osztályozást végezhesse.

A tanárok könnyen osztályozhatják és visszajelzést adhatnak a munkára a rugalmas rubrikák

segítségével. Az eredményeket egyenként visszaküldhetik a diákoknak, vagy exportálhatják azokat egy már meglévő évfolyamkönyvbe. A platform részletes elemzést kínál a munkáról, minden egyes hallgatóra, csoportra, valamint a kérdésekre adott válaszokra vonatkozóan is. Ez lehetővé teszi az oktatók számára, hogy pontosan megértsék a diákok teljesítményét és fejlődését a tanulási folyamat során.

2.1.3 Funkciók

A Gradescope platform támogatja a tesztek használatát, mely a leghatékonyabb és legkényelmesebb módot kínálja a gyors értékelésre. Ahogyan a 2.2. ábra is bemutatja, egyszerűen létrehozhat kérdés-válasz teszteket, melyekben a diákok az válaszok mellé rendelt betűket jelölik be. Ezeket könnyedén beolvashatja az alkalmazás segítségével, ami automatikusan felismeri és értékeli a válaszokat. A tanárok ellenőrizhetik az értékelést, mielőtt exportálják vagy elemzik az eredményeket.

2.2. ábra: Gradescope teszt létrehozása Atwood és Singh, *labeyearlabeyeardivisionlabelmonthlabelday*

Ez a pontozási rendszer kiválóan alkalmazható kódolási feladatoknál is, mivel képes automatikusan felismerni és pontozni a kódot a feltöltött adatok alapján. Ez hasonlóan működik a GitHub és a Bitbucket platformokhoz, lehetőséget adva a tanároknak, hogy szükség esetén kézi értékelést és visszajelzést is adjanak.

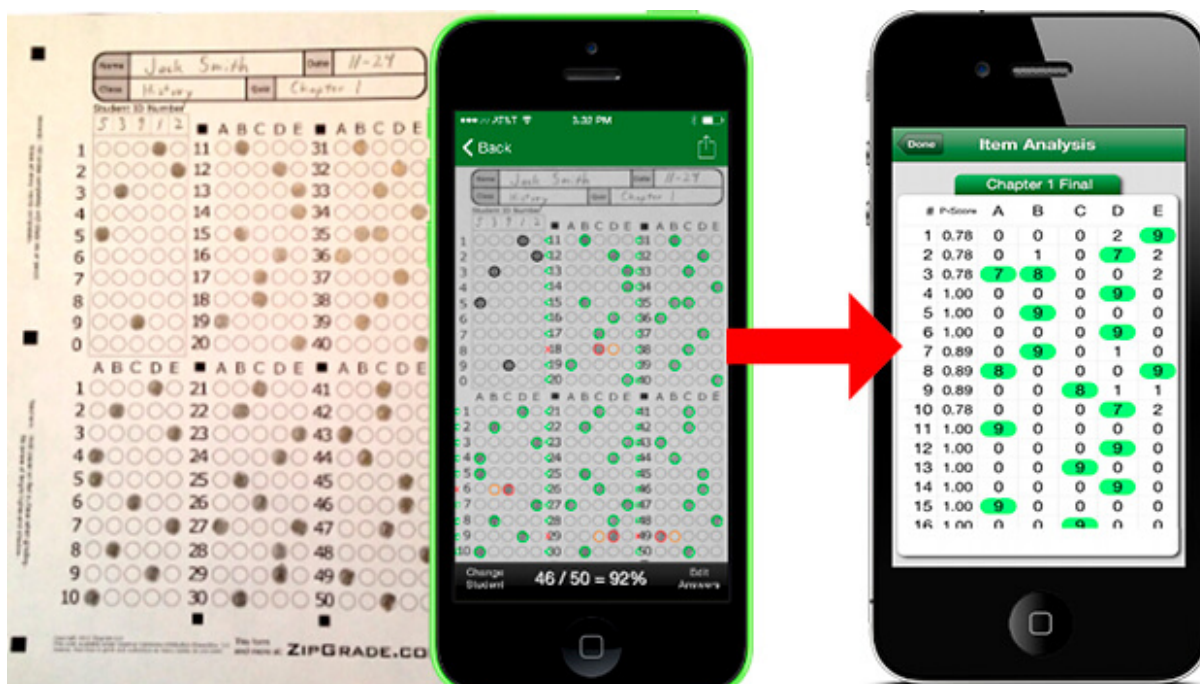
Az, hogy ez a szkennelésen alapuló értékelési rendszer alkalmazható vizsgákon is, jelentősen megkönnyíti a beadási és értékelési folyamatot. A digitalizáció lehetővé teszi, hogy könnyen hozzáférjünk az adatokhoz és elemezzük azokat.

2.2 ZipGrade

A ZipGrade alkalmazást a Surjono és tsai., labelyearlabelyeardivisionlabelmonthlabelday forrás-munkába alapján ismertetem. A legtöbb oktató számára az értékelési az értékelési tevékenységek túl adminisztratívák, mivel az eredményeket ki kell javítani és elemezni. Számos oktató a mai napig hagyományos kézi feladatsor összeállítást és pontozást alkalmaz, azonban ez rendkívül időigényes folyamat. Az oktatókat nagyban segíti a ZipGrade általi hatékony javítás.

2.2.1 Mi az a ZipGrade?

ZipGrade egy alkalmazás a válaszlapon történő javításra android és iOS eszközökön. Ahogyan a 2.3. ábrán is látható, lehetővé teszi a diákok számára, hogy válaszokat adjanak, fekete körrel kitöltve a választási lehetőségeket tollal, ceruzával vagy filctollal. A Gradescope működésétől eltérően nincs szükség számítógépes szkennelésre. ZipGrade, labelyearlabelyeardivisionlabel-monthlabelday



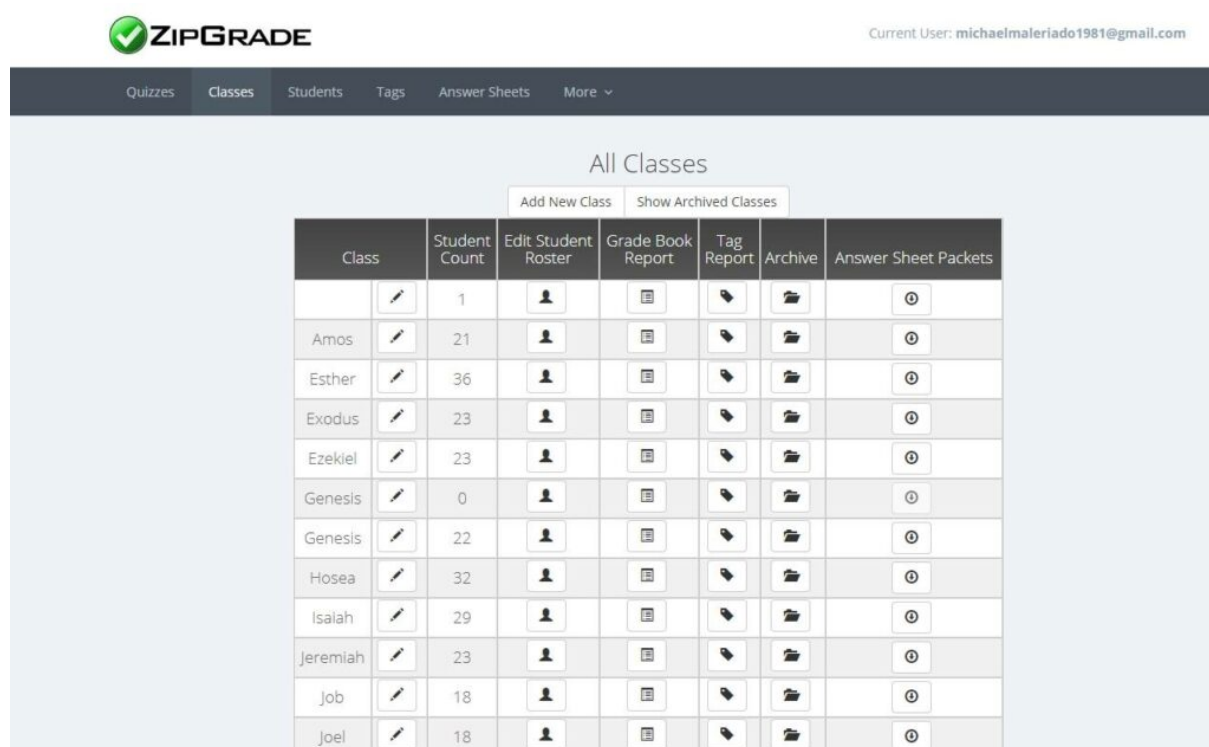
2.3. ábra: Zipgrade alkalmazás

2.2.2 Hogyan működik?

The Explorer, labelyearlabelyeardivisionlabelmonthlabelday Az alkalmazásban többféle válaszlappal érhető el: 20, 50 és 100 kérdéses válaszlappal, emellett lehetőségünk van egyedi válaszlappal létrehozására. A válaszok jelölésére használható toll, ceruza vagy filctoll, legfeljebb 3 betű választható ki egy-egy tesztkérdés esetén. A keresőket a válaszlapok fotózásakor a lap 4 sarkában elhelyezkedő négyzetre irányítva tudjuk osztályozni az adott diák feladatát, amiről azonnali összefoglalót kapunk. A kapott eredmények letölthetők PDF, illetve CSV formátumban. ZipGrade, labelyearlabelyeardivisionlabelmonthlabelday

2.2.3 Funkciók

Az alkalmazás letöltését és a regisztrációt követően létrehozhatunk a 2.4. ábrához hasonlóan saját osztályt, amelyhez manuálisan, illetve csv fájl importálásával is hozzáadhatjuk a diákokat. Ajánlott minden diákhoz egyedi Zipgrade ID-t rendelni, ugyanis beolvasáskor az eredmény rögtön a diák nevéhez kerül. Amennyiben csv fájl segítségével rendeljük hozzá a diákokat az osztályhoz, a Zipgrade automatikusan hozzárendel minden diákhoz egy egyedi azonosítót.



ZIPGRADE

Current User: michaelmariado1981@gmail.com

Quizzes Classes Students Tags Answer Sheets More

All Classes

Add New Class Show Archived Classes

Class	Student Count	Edit Student Roster	Grade Book Report	Tag Report	Archive	Answer Sheet Packets
	1					
Amos	21					
Esther	36					
Exodus	23					
Ezekiel	23					
Genesis	0					
Genesis	22					
Hosea	32					
Isalah	29					
Jeremiah	23					
Job	18					
Joel	18					

2.4. ábra: Saját osztály létrehozása

Elérhetőek válaszlap sablonok 20-tól 100 kérdéses verzióig, azonban készíthetünk egyéni válaszlapot is. A válaszlehetőségek megadása testre szabható, a végső kimenetet bármikor visszavonhatjuk, ha nem egyezik az elérni kívánt eredménnyel.

2.3 Konklúzió

Az általam tervezett alkalmazás sok hasonlóságot mutat a fentebb felsorolt alkalmazásokhoz a tesztlapok létrehozásának és azok értékelésének szempontjából, azonban bizonyos részeken igen eltérő. A Gradescope és a ZipGrade főként válaszlapok alapján történő értékelést támogatnak, azonban az általam tervezett alkalmazás lehetővé teszi az esszé feladatok létrehozását és értékelését is. Ez azt jelenti, hogy az oktatók könnyedén készíthetnek esszé feladatokat is a diákok számára, és az alkalmazás segítségével hatékonyan tudják értékelni és visszajelzést adni ezekre a feladatokra. A két fent említett alkalmazás a diákok által feltöltött válaszokkal és vizsgalapokkal dolgozik, ezzel szemben az én alkalmazásomban az oktatók vagy az adminisztratív dolgozók töltik fel a feladatlapokat a rendelkezésre álló szkennertől, ezzel kiszűrve a hibalehetőségeket a feltöltés részéről, mint például rossz fényviszonyok közt lefotózott vizsgalap, duplikációk feltöltése, más munkájának leadása. Emellett az általam készített alkalmazás lehetőséget nyújt arra, hogy a vizsgákat több oktató is értékelhesse egyszerre, ezzel is gyorsítva a javítás folyamatát.

3 FELHASZNÁLHATÓ KÉPFELDOLGOZÓ ELJÁRÁSOK

A képfeldolgozási folyamatok és műveletek mélyebb megismerése elengedhetetlen a feladatom megvalósításához. A rendszernek képesnek kell lennie a papíralapú vizsgák automatikus feldolgozására, a dokumentumok egyenesítésére és előkészítésére a további feldolgozási lépésekhez, mint például a karakterfelismerés. A rendszer központi eleme egy olyan algoritmus lesz, amely automatikusan azonosítja a vizsgalapokon található két legnagyobb négyszöget, majd ezen négyszögek alapján forgatja és igazítja a dokumentumokat, valamint kivágja az oktató számára releváns részeket. A vizsgalapok felépítését a 3.1. ábra mutatja be:

VIZSGA AZONOSÍTÓ	T	Z	A	9	6	0
NEPTUNKÓD						
NÉV						

FELADAT AZONOSÍTÓ	T	Z	A	9	1	3
Melyik UML diagram nem viselkedést ábrázol?						
A. use-case B. state machine C. timing D. class						
A helyes válasz betűjele:						
CQIS esetén melyik ritmizálható el optimalizációs ciklusból a query előadás?						
A. left B. if C. data-access D. database						
A helyes válasz betűjele:						
Mi nem igaz a Service Layer?						
A. Feladata lehet a tranzakció kezdése is B. Hírdetést fogad és továbbít a hálón C. SOA felületet valósít meg D. UI nem lehet a része						
A helyes válasz betűjele:						
CQIS esetén melyik fájl/rendszert nem szerepel a command előadás?						
A. Event Store B. Event Bus C. Command Bus D. Persistent View Model						
A helyes válasz betűjele:						
Mi nem igaz az MQTT protokollról?						
A. ISO szabvány B. Ki overload C. Publish-Subscribe (szemléltetőre való) D. Lassabb, mint a HTTP						
A helyes válasz betűjele:						
Mi nem igaz a mikroszolgáltatásokról?						
A. Össző alkalmazások B. Csak domain szinten ábrázolhatók C. Előre nyelven írhatók D. Ismeretlen példák kommunikáció						
A helyes válasz betűjele:						
Ügyeljen arra, hogy a válaszok betűjeleit úgy írja be, hogy azok ne lógjanak ki a keretből.						

VIZSGA AZONOSÍTÓ	T	Z	A	9	6	0
NEPTUNKÓD						
NÉV						

FELADAT AZONOSÍTÓ	T	Z	A	9	1	1
Pontszám: 10						
Modellezze az alábbi működési időtérítést (timing diagram) segítségével!						
A feladatnál a blokkjelölés megnevezése a következő: (https://en.wikipedia.org/wiki/Block_diagram). A DMC kártya nem fut le, mert a blokkjelölés cache-ében megvan az IP cím. Mivel a feladatnál 3rd party bejelentkezési hálózati (facebook login), a JWT tokenek a HTTP GET kérésrel együtt érkeznek a webalkalmazásba (100ms). A webalkalmazás szerveroldali HTTP GET kérés küld a facebook.com oldalnak, hogy validálja a JWT tokenek. A kérés kb. 50ms alatt kimegy, a válasz kb. 1500ms alatt érkezik meg. A validáció eredményeként vagy hibázott a token meg, vagy a bejelentkezési megkapja a tokenet (idő: 100ms). A feladatnál a blokkjelölés a következő: (https://en.wikipedia.org/wiki/Block_diagram). A kérés kb. 500ms alatt kimegy.						

VIZSGA AZONOSÍTÓ	T	Z	A	9	6	0
NEPTUNKÓD						
NÉV						

FELADAT AZONOSÍTÓ	T	Z	A	9	1	2
Pontszám: 2						
A projektmenedzsment vezérlőrendszer c. ábrán milyen 3 jellemző ábrázolunk?						

FELADAT AZONOSÍTÓ	T	Z	A	9	6	4
Pontszám: 3						
Mire való a Visitor tervezési minta?						

FELADAT AZONOSÍTÓ	T	Z	A	9	6	5
Pontszám: 1						
Sorolja fel a SOLID elveket!						

3.1. ábra: Feladatok sablonjai

- Vizsga azonosító: A lap tetején, a bal oldalon található, egy hat rubrikából álló téglalapban szükséges megadni.
- Neptun kód: A vizsga azonosítótól jobbra található egy újabb téglalap, amely szintén hat üres rubrikát tartalmaz, ahova a hallgatónak a saját azonosítóját kell beírnia nyomtatott nagybetűkkel. A rubrikák célja a karakterek elválasztásának megkönnyítése a későbbi karakterfelismeréshez.
- Név: A vizsga azonosító és a Neptun kód alatti részen található egy üres mező, amely a hallgató nevét kéri.
- Feladat azonosító: A feladat azonosítókat közvetlenül a vizsgafeladatok blokkja felett, bal oldalon, hat karakterből álló rubrikákba kell beírni. Az azonosítók a feladat típusától függően kerülnek megadásra: az esszé típusú feladatok esetében az azonosító hatodik karaktere 1, a rövid kérdéseké 2, a tesztfeladatoké pedig 3. Az ötödik karakter az azonos

típusú feladatok sorszámát jelzi, például az első esszé feladat azonosítója: XXXX11, a másodiké XXXX21.

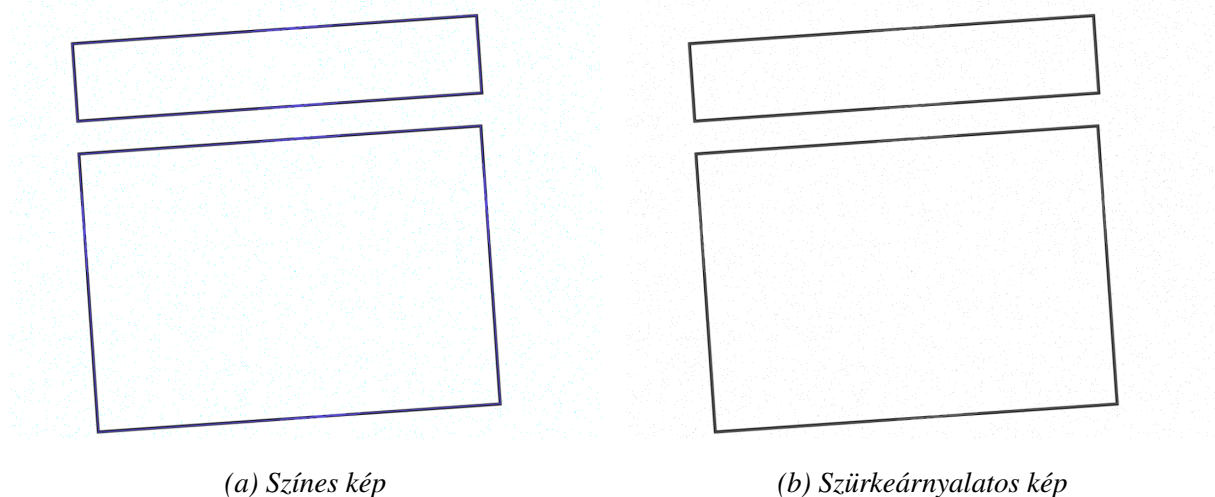
3.1 Képek előfeldolgozása

Az egyik első lépés a vizsgalapok szkennelt képeinek előfeldolgozása, amely biztosítja a megfelelő alapot a későbbi algoritmusok számára. A szkennelés során keletkező képeken gyakran előfordul, hogy a lapok nem egyenesek, a kép torzul, ezért fontos a dokumentumok megfelelő mértékű elforgatása/egyenesítése, illetve a kép minőségének javítása.

3.1.1 Kép elforgatása

A vizsgalapok feldolgozása során a szkennelt képek elforgatására és egyenesítésére egy kezdeti lépésként a Canny élkereső algoritmus alkalmazása kerül előtérbe. Ez lehetővé teszi, hogy a vizsgalapokon pontosan azonosítsuk a kontúrokat, minimalizálva a zajból vagy torzításból eredő hamis éldetektálásokat. Az algoritmus feladata a vizsgalapokon található két legnagyobb néyszög automatikus azonosítása, majd ezek alapján a dokumentumok megfelelő orientációjának és elrendezésének beállítása. Az algoritmus az alábbi fő lépéseket követi Canny, labelyearlabelmonthlabelday:

- Kép szürkeárnyalatossá konvertálása: az eredeti színes képen minden pixelhez hozzárendelt RGB (vagy RGBA) értékek átalakulnak egyetlen intenzitási értéké, amely jelzi a szürkeárnyalatot. Ez az intenzitás érték általában egy súlyozott átlag vagy más transzformáció eredménye. Példa egy kép szürkeárnyalatossá konvertálásáról (3.2. ábra):



3.2. ábra

- Zajcsökkentés: A szkennelt képet Gauss-szűrővel simítjuk, hogy csökkentsük a zaj hatását. A Gauss-szűrés egy aluláteresztő szűrő, mely csökkenti a véletlen zajt a képen, de megőrzi a képen lévő éleket.
- Gradiensek számítása (Sobel-operátor): Az intenzitásváltozások detektálásához a szűrt képen kiszámítjuk a gradienseket. Ehhez a Sobel-operátort használjuk, amely két irányban (vízszintes és függőleges) azonosítja az éleket. A kép különböző pontjaiban a gradiens magnitúdója (amely a térbeli változás mértékét mutatja meg egy adott pontban) és iránya alapján meghatározhatók azok a területek, ahol intenzitásváltozás történik, vagyis ahol élek találhatók.

A gradiens értékei az alábbi képletekkel számíthatók ki Kanopoulos és tsai., labelyearlabelmonthlabelday:

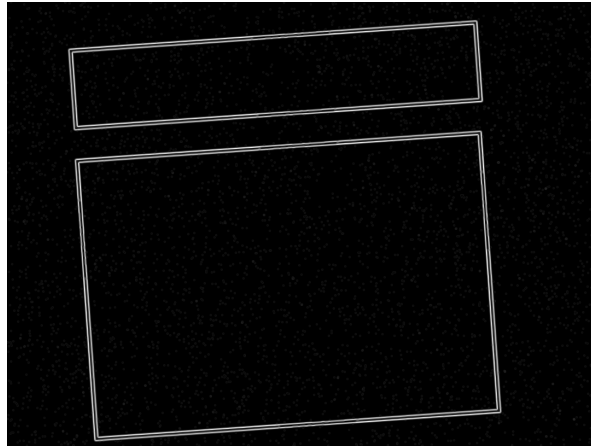
G_x = Sobel operátor vízszintes irányban, G_y = Sobel operátor függőleges irányban

A gradiens magnitúdója:

$$|\nabla f(x, y)| = \sqrt{G_x^2 + G_y^2}$$

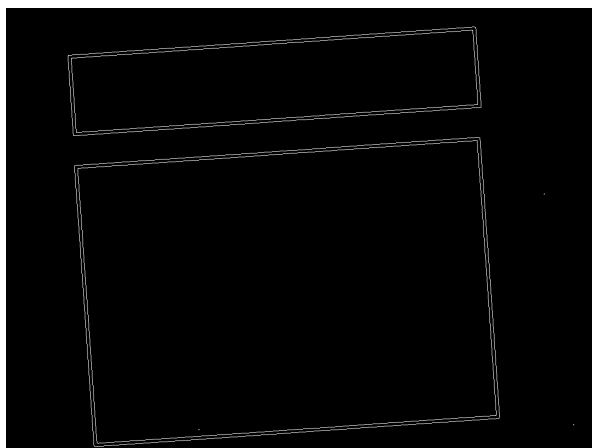
Ezt követően minden olyan képpontot, amelynek gradiens értéke egy adott küszöbérték felett van, fehérre (1) állítunk, a többit pedig feketére (0), ezt a 3.3. ábra szemlélteti:

$$I_{\text{bin}}(x, y) = \begin{cases} 1, & \text{ha } |\nabla f(x, y)| > T \\ 0, & \text{ha } |\nabla f(x, y)| \leq T \end{cases}$$



3.3. ábra: A Sobel-operátor alkalmazásának eredménye

- Non-maximum suppression (NMS): A kép magnitúdója vastag éleket hozhat létre. A kívánt végeredmény vékony éleket igényel, ezért Non-maximum suppression-t kell alkalmaznunk az élek vékonyítása érdekében. A gradiens irányainak és magnitúdójának számítását követően a nem releváns éldetektált pontokat elnyomják. Ez a lépés biztosítja, hogy csak a maximális gradiensértékek maradjanak meg, amelyek az igazi élekhez tartoznak. A Non-maximum suppression olyan módon működik, hogy az él irányában vizsgálja meg a szomszédos pixeleket, és azokat a pontokat, amelyek nem lokális maximumpontok, elnyomja.
- Kettős küszöbölés: Az algoritmus végső lépéseként a pixelek gradiensértéke alapján kettős küszöbölést alkalmazunk, amely egy magas és egy alacsony küszöbértéket határoz meg. A magas küszöb feletti értékek biztos éldetektálásnak számítanak, míg az alacsony küszöb közötti értékek akkor maradnak meg, ha a közelben találhatóak olyan pixelek, amelyek biztos élekhez tartoznak. Ezt nevezzük hiszterézis alapú élkövetésnek, amely elkerüli a szaggatott élvonalak kialakulását. A végeredmény a 3.4. ábrán látható.



3.4. ábra: Canny éldetektáló algoritmus végeredménye

A két legnagyobb téglalap pontos felismerése és befoglalása alapvető fontosságú a minél pontosabb elforgatáshoz, így egy fő szempont, hogy ferde téglalapokat is fel tudjunk ismerni. Az OpenCV az egyik legnépszerűbb és legszélesebb körben használt számítógépes látási könyvtár, ezért egy aktív közösség támogatja. Ez gyors válaszokat és rengeteg erőforrást jelenthet a problémák megoldásában, valamint a projektek gyors előrehaladásában. Emellett a függvényei optimalizáltak és jól dokumentáltak, ami gyors fejlesztési ciklust tesz lehetővé, így ezen könyvtár függvényeit vizsgáltam meg a feladat elvégzéséhez: `Developers`, `label`, `year`, `label`, `year`, `di`, `vision`, `label`, `month`, `label`, `day`.

- A `boundingRect()` függvény egy egyszerű és gyors megoldás, amely a bemeneti kontúrhoz igazított legkisebb, tengelyekhez igazított téglalapot számítja ki. Mivel csak az x- és y-tengelyekhez illeszkedő téglalapot keres, elforgatott alakzatok esetén nem nyújt pontos megoldást a legkisebb lehetséges téglalapra, csak egy olyan befoglaló téglalapot kapunk, amely nem forgatható el. Egyetlen bemenete egy kontúr vagy pontok halmaza, amit gyakran `cv::Point` típusú vektorral adunk meg. A kimenet egy tengelyekhez igazított (axis-aligned) téglalap, amely négy int értéket tartalmaz: (x, y, width, height). Ezek a következők:

- x: a téglalap bal felső sarkának x-koordinátája
- y: a téglalap bal felső sarkának y-koordinátája
- width: a téglalap szélessége
- height: a téglalap magassága

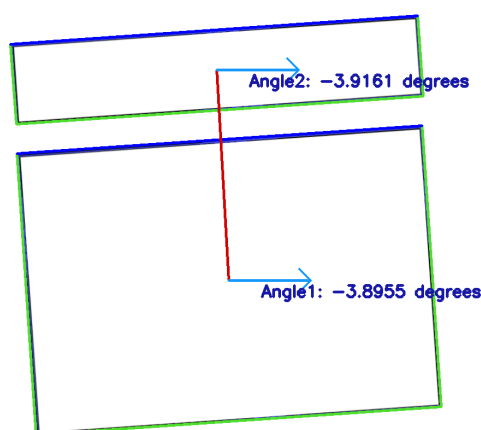
A számítás gyors és egyszerű, de csak vízszintes és függőleges irányban illeszkedik. Emiatt a `boundingRect()` függvény nem ideális választás a feladat szempontjából.

- A `minAreaRect()` egy fejlettebb függvény, amely a legkisebb területű forgatott téglalapot adja vissza, amely teljesen befoglalja a kontúr. Ez különösen akkor hasznos, ha az objektumok elforgatottak, vagy az alakzatok pozíciója változatos. Bár számításigényesebb, pontosabb eredményt ad, mivel a téglalap orientációját is figyelembe veszi. A függvény visszaadja a téglalap középpontját, méreteit és forgatási szögét, ami lehetővé teszi a rugalmasabb objektumkezelést, így ez a függvény jó választás lehet a ferde kép okozta probléma kiküszöböléséhez. A bemenet egy kontúr vagy pontok halmaza, amit szintén egy `cv::Point` típusú vektorral adhatunk meg. A függvény kimenete egy `RotatedRect` típusú objektum, amely három fontos paramétert tartalmaz:

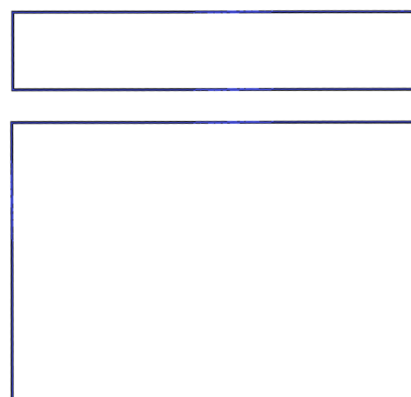
- `center`: a téglalap középpontjának koordinátái (`cx`, `cy`) formátumban.
- `size`: a téglalap mérete (`width`, `height`) formátumban.
- `angle`: a visszaadott szög értelmezése (fokban mérve) a téglalap hosszabb oldalához igazodik az x-tengelyhez képest, azaz a szög értéke azt mutatja, hogy a téglalap hosszabb oldala mennyire tér el az x-tengely irányától, az óramutató járásával megegyező irányban.

Mivel a `minAreaRect()` által visszaadott `angle` a téglalap hosszabb oldalához igazodik az x-tengelyhez képest, elég kivonnunk belőle 90 fokot, hogy megkapjuk a teljes függőlegeshez szükséges korrekciót.

A 3.5. (a) ábra szemlélteti a forgatási szög kiszámítását, a 3.5. (b) ábra pedig a forgatás eredményét:



(a) Átlagos dőlésszög kiszámítása

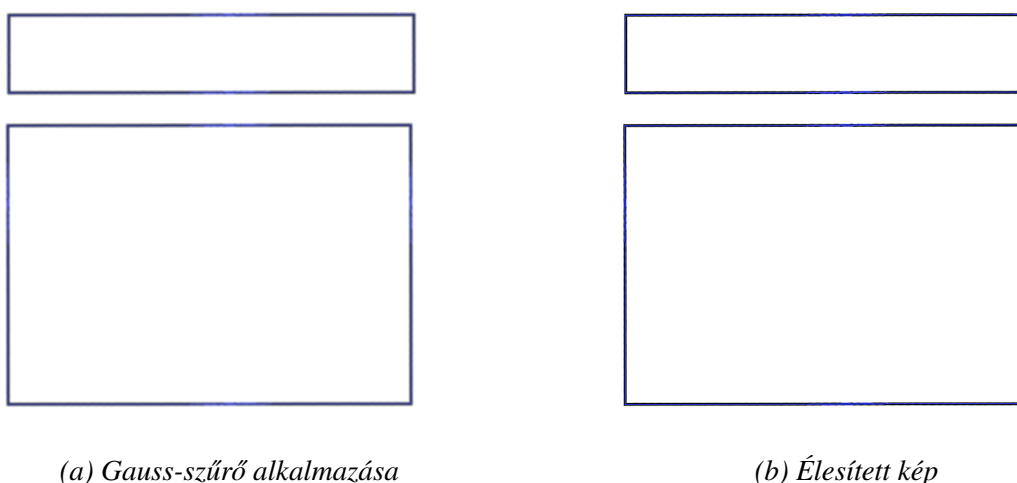


(b) Kép forgatása a dőlésszög alapján

3.5. ábra

3.1.2 Releváns részek kivágása

A számunkra releváns, feladatot tartalmazó téglalap kivágása előtt javítanunk kell a kép minőségén. A zajcsökkentés elősegíti az algoritmusok hatékonyabb működését, mivel kevesebb torzított vagy nem kívánt információt tartalmazó képet tudunk előállítani, ezzel csökkentve a hibás adatfeldolgozás esélyét. Erre használhatjuk a fentebb említett Gauss-szűrőt. A Gauss-szűrő minden képpont értékét módosítja egy súlyozott átlagolás alapján, ahol a távolabbi pontok kisebb súlyt kapnak. Ennek eredményeként a szűrt kép simább lesz, a zajos területek elsimítódnak, miközben a fontosabb élek élesek maradnak, így megőrződik a kép olvashatósága és részletessége. Atawia és tsai., labelyearlabelyeardivisionlabelmonthlabelday Az AddWeighted() függvény segítségével az eredeti kép és a Gauss-szűrővel elmosott kép kombinálása élesebb kontúrokat eredményez, mivel az elmosott képet negatív súllyal kivonhatjuk az eredeti képből. Ez a folyamat - amelyet a 3.6. ábra mutat be - javítja a kép élességét és kontrasztját, elősegítve a további feldolgozási lépések, mint például az elmosódás-csökkentés és a kontrasztjavítás hatékonyságát. Documentation, labelyearlabelyeardivisionlabelmonthlabelday



3.6. ábra

Ezt követően a képből az alábbi információkat szükséges kinyerni: a válaszadók Neptun kódját, a feladat azonosítóját, valamint a vizsgakérdésre adott válaszokat kép formátumban. Teszt es etén a válaszok mellett az elért pontszámot is meg kell határozni. A vizsgafeladatok formátuma előre meghatározott, összeállításuk a rendelkezésre álló minták alapján történik, így előre behatárolható, hogy mely területeken érdemes keresni az egyes információkat, elkerülve ezzel az egész képre kiterjedő, időigényes vizsgálatot. A Neptun kódok és a feladatok azonosítói

jellegzetesen hat négyzetes rubrikában helyezkednek el, ezeket kell elsődlegesen detektálnunk a későbbi karakterfelismerés céljából. A feladathoz rendelt azonosítók egyértelmű iránymutatást adnak a vizsgakérdés típusáról, lehetővé téve, hogy a képen célzottan kutassunk az adott kérdést tartalmazó téglalapok után. Tesztfeladat esetén a válaszadás formája meghatározott: a helyes válasz betűjelét négyzetekbe szükséges írni. Ennek megfelelően a feldolgozás során ezen négyzetek detektálása és a sorrendiség szerinti összeállítása szükséges, hogy egyetlen képpé kombinálva ki tudjuk nyerni a válaszokat. A kinyert válaszokat ezt követően össze kell vetnünk az előzetesen rögzített helyes válaszokkal, biztosítva a pontos kiértékelést.

3.1.3 Értékelés

A Canny algoritmus egyik legnagyobb erőssége, hogy több lépcsős folyamatot alkalmaz, beleértve a zajszűrést, gradiens számítását és a hiszterézis alapú élkövetést, amely elkerüli a szaggatott élek létrejöttét. Más élkereső algoritmusok, mint például a Sobel vagy a Prewitt, egyszerűbbek és gyorsabbak lehetnek, azonban ezek gyakran kevésbé hatékonyak a zajos vagy torz képeken. A Sobel és Prewitt algoritmusoknál nem alkalmaznak hiszterézis alapú élkövetést, így ezek hajlamosak lehetnek szaggatott éleket detektálni, különösen bonyolultabb, zajos képeken. Továbbá, bár a Canny algoritmus jól teljesít az élek pontos azonosításában, az algoritmus paraméterezése (például az alacsony és a magas küszöbértékek beállítása) kihívást jelenthet, mivel a paraméterek helytelen megválasztása nem kívánt éldetektálásokat vagy az élek hiányát eredményezheti. Jose és tsai., labelyearlabelyeardivisionlabelmonthlabelday

Gyenge minőségű vagy torzult képeken jelentős pontosságcsökkenés következhet be. Amennyiben a szkennelt képek zajosak, homályosak vagy torzítottak, akkor az algoritmusnak nehézségei lehetnek az élek pontos detektálásában és a dokumentumok megfelelő igazításában. Az ilyen képek feldolgozása során a Canny élkereső zajcsökkentése ellenére is gyakrabban előfordulhatnak hibás éldetektálások, különösen, ha a szkennelési minőség jelentősen csökken. Egy gyenge minőségű szkennelés során előfordulhat, hogy a dokumentum élei elmosódnak, vagy a háttérzaj túl erős ahhoz, hogy az algoritmus helyesen detektálja az éleket. Ebben az esetben a dokumentum egyenesítése pontatlan lehet, ami az egész feldolgozási folyamatra hatással lehet, beleértve az karakterfelismerést is.

Ezen probléma kiküszöbölésére alkalmazható a medián szűrés, illetve az adaptív küszöbölés. A medián szűrés megőrzi az éleket, mivel minden pixel értékét a környezetében található

pixelek mediánja alapján állítja be, így csökkentve a zajos pontok hatását, de megtartva a kép éles kontúrjait. Az adaptív küszöbölés pedig dinamikusan alkalmazza a küszöbértéket a kép különböző részeire, így jobban alkalmazkodik a változó fényviszonyokhoz, és képes hatékonyan elkülöníteni a zajt a fontos képi elemektől, különösen, ha az egyenletlen megvilágítás problémát okoz. Cao és Liu, labelyearlabelyeardivisionlabelmonthlabelday

3.2 Karakterfelismerés

A karakterfelismerés egy igen fontos pont a rendszer helyes működésének szempontjából. A vizsga és feladat azonosító kódok alapján létrehozott mapparendszerek, illetve az automatikusan kijavított tesztfeladatok pontozása ezen alapul.

3.2.1 Tesseract OCR

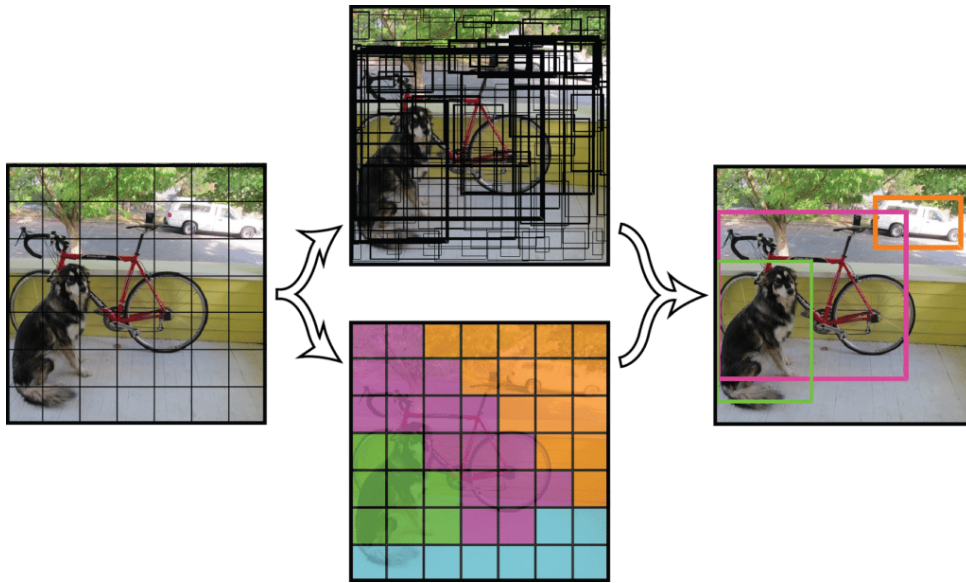
Az OCR az "Optical Character Recognition", azaz "optikai karakterfelismerés" rövidítése. Ez egy olyan technológia, amely felismeri a digitális képen belüli szöveget, és általában a beolvasott dokumentumok és képek szövegének felismerésére használják. A Tesseract egy nyílt forráskódú, számos nyelvet támogató szövegfelismerő motor, amely az Apache 2.0 licenz alatt érhető el. Használható közvetlenül, vagy API használatával, nyomtatott szöveg kinyerésére a képekből. Kiválóan alkalmazható nyomtatott szöveg felismerésére, azonban kézírást felismerésére nem a legalkalmasabb módszer.

A Tesseract a gépirásos szövegek esetében 90-95%-os pontosságot ért el, kézírástos szövegek esetén azonban a teljesítménye jelentősen csökken, a pontosság 70-80%-ra eshet vissza. A Tesseract karakterhibaaránya (CER) kézírástos szövegek esetén 20-30%-ra is nőhet. Smith, labelyearlabelyeardivisionlabelmonthlabelday

3.2.2 Yolo algoritmus

Redmon és tsai., labelyearlabelyeardivisionlabelmonthlabelday A You Only Look Once (YO-LO) egy nyílt forráskódú, korszerű objektumfelismerő, neurális hálózatokon alapuló algoritmus, amely objektumdetektálást tesz lehetővé valós időben. Nevét a „You Only Look Once” angol kifejezésből kapta, ami azt jelképezi, hogy egyetlen algoritmus futás alatt képes különböző objektumok detektálására. Népszerűségét elsősorban a sebességének és pontosságának köszönheti.

A 3.7. ábra bemutatja a felismerés folyamatát:



3.7. ábra: Felismerés folyamata

Első lépésben az eredeti képet (A) egyenlő alakú $N \times N$ rácscellákra osztjuk. A rács minden cellája felelős a lefedett objektum osztályának lokalizálásáért és előrejelzéséért, valamint a valószínűség/megbízhatóság értékéért. A következő lépés a határolókeretek meghatározása, amelyek megfelelnek a kép összes objektumát kiemelő téglalapoknak. Annyi határolókeretünk lehet, ahány objektum van egy adott képen. A kép egyetlen objektumának legtöbbször több rácsdoboz-jelöltje is lehet az előrejelzéshez, még akkor is, ha nem mindegyik releváns. Az IOU (Intersection over Union) egy metrika az objektumérzékelési feladatokban, amely az objektumkeretek átfedésének mértékét méri a valósághoz képest és célja az ilyen rácsdobozok elvetése, hogy csak a relevánsak maradjanak meg. Az IOU küszöbértékének beállítása nem mindig elegendő, mert egy objektumnak több doboza is lehet, amelyek IOU-ja meghaladja a küszöbértéket, és az összes mező elhagyása zajt tartalmazhat. Ekkor alkalmazhatjuk az NMS-t, hogy csak a legnagyobb valószínűségű észlelési pontszámmal rendelkező dobozokat tartsuk meg.

Kézírtas karakterek azonosítása során a YOLO nem nyújtott megfelelő teljesítményt, és a pontosság jelentősen csökkent, jellemzően 50% alatti eredményeket mutatott, különösen a változatos írásmódok miatt. Mivel a projekt egyik célja a kézírtas számjegyek és az angol ábécé nagybetűs karaktereinek felismerése, úgy döntöttem, hogy kipróbálom a YOLO algoritmust, mivel feltételeztem, hogy a nyomtatott nagybetűs kézírtas karakterek esetleg

könnyebben felismerhetők számára. Azonban a modell pontossága ezen a feladaton sem sok esetben haladta meg az 50%-ot.

Bár a YOLO egy rendkívül hatékony objektumfelismerési algoritmus, amely valós időben képes detektálni különböző objektumokat, nem mindig ideális a kézírással írt karakterek, különösen változatos írásmódok esetén.

3.2.3 Transformer-alapú Optikai Karakterfelismerés

A Transformer-based Optical Character Recognition (TrOCR) egy encoder-decoder architektúrát alkalmaz, ahol a képi bemenetet egy képi Transformer kódolja, míg a kimenet dekódolásáért egy szöveg Transformer felelős. Az encoder és a decoder előképzett modellekből származik, mint például a BEiT (a képekhez) és a RoBERTa (a szövegekhez). A TrOCR modellek előtrénin-gezése két fázisra oszlik: Az első fázisban egy nagy mennyiségű, nyomtatott szöveget tartalmazó adatbázison tanítják a modellt. Ebben a fázisban a modell elsajátítja a vizuális jellemzők ki-nyerését és a karakterek azonosítását. A második fázisban a modell finomhangolása történik a kézírással írt szövegekhez specifikus adatbázisok használatával. A cél, hogy a modell képes legyen a kézírással írt karakterek azonosítására is, nemcsak a nyomtatott szövegekre. A kutatók külön figyelmet fordítanak a kézírással írt szövegek variabilitására, mivel a kézírással írt karakterek gyakran eltérnek a nyomtatott formátumtól. A TrOCR teljesítményét különböző adathalmazon tesztel-ték, beleértve az IAM kézírással írt adatbázist. A kísérletek során a TrOCR kiemelkedő pontosságot mutatott, és a legjobbnak bizonyult a kézírással írt szövegfelismerésben a hagyományos OCR mo-dellekhez képest. Az eredmények alapján a TrOCR modellek jelentős fejlődést mutatnak az optikai karakterfelismerés terén, különösen a kézírással írt szövegek feldolgozásában. Li és tsai., labelyearlabelyeardivisionlabelmonthlabelday

A BEiT egy mélytanulós képfeldolgozó modell, amely a Transformer architektúrát alkal-mazza a képek jellemzőinek kinyerésére. Az alábbiakban a BEiT-tel kapcsolatos fontosabb információk találhatók Bao és tsai., labelyearlabelyeardivisionlabelmonthlabelday:

- Kép Tokenizálás: A bemeneti képeket 16x16 pixeles patch-ekre osztják, amelyek lineári-san beágyazódnak egy vektortérbe. Ez a folyamat lehetővé teszi a modell számára, hogy figyelembe vegye a képen belüli összefüggéseket.
- Masked Image Modeling (MIM): A BEiT során a képekből véletlenszerűen eltávolítanak

patch-eket, és a modellnek előre kell jeleznie a hiányzó információkat. Ez a megközelítés hasonló a masked language modeling (MLM) technikához.

- Alkalmazás: A BEiT kiválóan alkalmazható különféle képfeldolgozási feladatokhoz, mint például a képosztályozás és az OCR.

A RoBERTa a BERT továbbfejlesztett változata, amely a nyelvi modellezés során figyelembe veszi a BERT korlátait Liu és tsai., labelyearlabelyeardivisionlabelmonthlabelday:

- Optimalizáció: A RoBERTa eltávolítja a "next sentence prediction" (következő mondat előrejelzése) feladatot, és kizárólag a masked language modeling technikát használja. Ez növeli a modell hatékonyságát és pontosságát.
- Pre-training: A RoBERTa modellek a tanítási adatok méretének és változatosságának növelésével optimalizálják a teljesítményt. Az optimalizálás érdekében különféle hyperparamétereket is finomhangolnak.
- Alkalmazások: A RoBERTa széleskörűen alkalmazható különböző nyelvi feldolgozási feladatokhoz, beleértve a szövegklasszifikálást és a kérdés-válasz rendszereket.

A TrOCR a kézírási karakterek azonosításában 98%-os pontosságot ért el az IAM kézírási adatbázison. A karakterhibaarány körülbelül 3% körüli értéket mutatott, ami azt jelzi, hogy a legtöbb karaktert helyesen azonosította. Li és tsai., labelyearlabelyeardivisionlabelmonthlabelday

3.3 Konklúzió

A TrOCR modell, jelentős előrelépést ért el az optikai karakterfelismerés terén, különösen a kézírási karakterek azonosításában. A hibás karakterfelismerés torzíthatja a diákok teljesítményének értékelését, így a megbízható eredmények elérése különösen fontos.

Bár a Tesseract és a YOLO széles körben használt megoldások, nem a legmegfelelőbbek a kézírási karakterek feldolgozására. A Tesseract elsősorban nyomtatott, gépiírási szövegekre optimalizált, a YOLO pedig nem mindig képes megbirkózni a kézírási szövegek sokszínűségével és eltérő karakterisztikáival kapcsolatos kihívásokkal. Ezzel szemben a TrOCR mélytanulási megközelítése és a Transformer architektúra lehetővé teszi a kézírási szövegek variabilitásának kezelését, így a pontosabb és megbízhatóbb eredmények elérését.

Összességében a TrOCR egy ígéretes megoldás, amely lehetővé teszi a vizsgák információinak hatékony és pontos feldolgozását. További előnye, hogy saját adatokkal, saját adatbázissal is

finomhangolhatjuk a modellt, ezzel is növelve a későbbiekben a karakterfelismerés pontosságát.

4 RENDSZERTERV

Az alábbiakban ismertetem a projektem alkalmazásainak rendszertervét.

4.1 Szkennelő alkalmazás: Neptun kódok és szkennelt vizsgafeladatok kezelése

Az irodai adminisztrátorok által kezelt szkennelő alkalmazás egy WPF alkalmazás, amely lehetővé teszi a feldolgozott adatok feltöltését a szerverre. Ezen a szerveren keresztül az oktatók könnyedén elérhetik és megtekinthetik az adatokat.

4.1.1 Cél

Az adminisztrátorok által kezelt szkennelő alkalmazás egy WPF-alapú felület, amely lehetővé teszi a beszkennelt vizsgalapok adatainak feldolgozását és feltöltését a szerverre. A fő funkciók között szerepel a beszkennelt képek beolvasása, a Neptun kódok és feladattípusok és megoldások automatikus azonosítása, a releváns részek körbevágása, valamint az adatok PostgreSQL adatbázisba történő mentése REST API-n keresztül. Az oktatók így közvetlenül elérhetik a diákok megoldásait a szerveren keresztül.

4.1.2 Funkcionális követelmények

- Automatikus feladat azonosító, Neptun kód, valamint - tesztkérdések esetén - válasz felismerés.
- Oktatói jogosultságok beállítása adott vizsgához.
- Kérdések és válaszok körbevágása a felesleges területek minimalizálása érdekében.

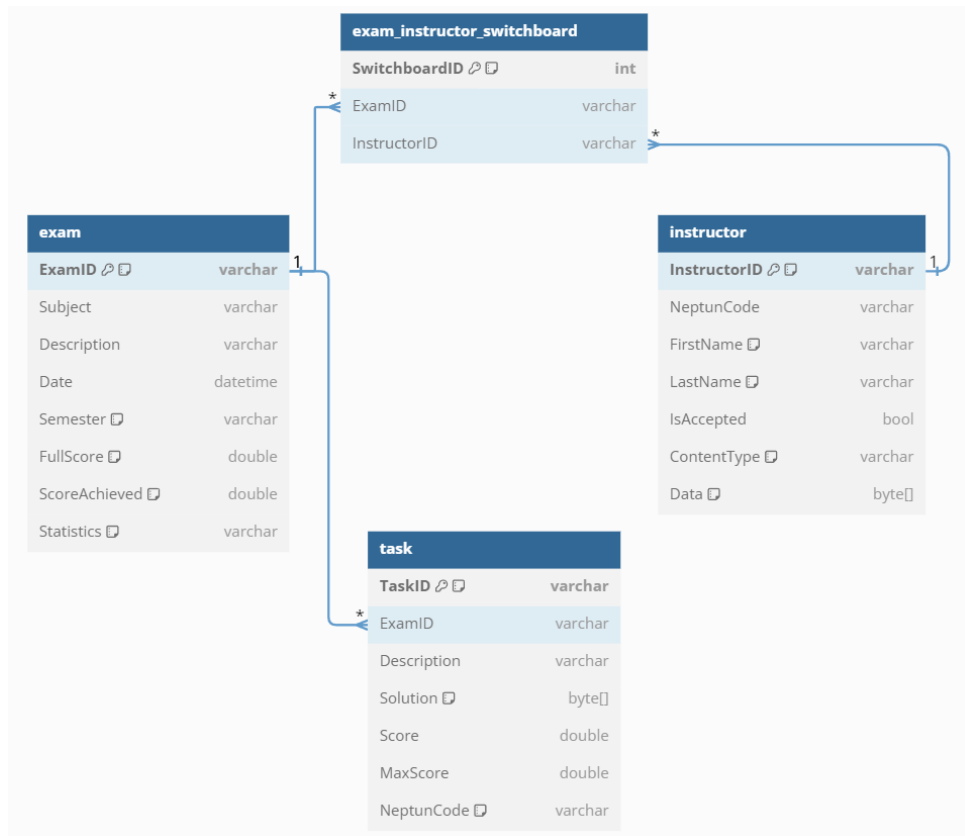
4.2 Szerveroldali és -javító alkalmazás

A szerveroldali alkalmazásom a különböző végpontokat kezeli. Az alkalmazás felépítése:

- **Kontrollerek (Controllers):** A kontrollerek felelősek az endpointok kezeléséért, azaz fogadják a kliensalkalmazástól érkező kéréseket, és meghívják a szükséges üzleti logikát a kérések teljesítéséhez.
- **Üzleti logika (Business Logic):** Az üzleti logika az alkalmazás működésének fő része,

amely az adatfeldolgozást és a különböző műveletek végrehajtását végzi.

- **Adattár (Repository):** A repository réteg az adatbázis helyes működéséért és a vele való kommunikációért felelős, ide tartoznak az adatlekérdezések, mentések, törlések és módosítások. A tervezett adatbázis táblákat és kapcsolatokat a 4.2. ábra szemlélteti.



4.1. ábra: Táblák és kapcsolatok

A javító szoftver egy olyan alkalmazás, amelybe az oktatók az O365-ös címükkel léphetnek be. Lehetővé teszi a felhasználók számára, hogy könnyen és biztonságosan hozzáférjenek az alkalmazáshoz az általuk használt Office 365 bejelentkezési adataikkal. A bejelentkezett oktatók számára láthatóvá válnak a javítandó vizsgák, melyekhez hozzárendelték Őket, illetve az adminisztrátori jogosultsággal rendelkező felhasználók további jogosultságot adhatnak más oktatók számára a gyorsabb és hatékonyabb javítás érdekében. Az oktatók az értékelt vizsgákhoz megjegyzéseket is fűzhetnek, ezzel segítve a hallgatókat a fejlődésben és megértésben.

4.2.1 Cél

Az alkalmazásban az adminisztrátor szerepkörű oktatók lehetőséget kapnak mások adminisztrátori jogainak megadására, ahogyan a 4.2-es ábrán is látható. Ezáltal az adminisztrátorok hatékonyan kezelhetik a felhasználói jogosultságokat, és biztosíthatják, hogy az alkalmazásban csak az engedélyezett személyek férjenek hozzá a megfelelő funkciókhoz és adatokhoz.

Ezenkívül az adminisztrátorok képesek új vizsgák hozzáadására a rendszerhez, így biztosítva az alkalmazás folyamatos frissítését és a legújabb vizsgaadatok elérhetőségét. Az új vizsgák hozzáadásakor az alkalmazás lehetőséget biztosít az adott vizsga azonosítójának, nevének, dátumának és a résztvevő hallgatók neptun kódjainak megadására. Ezáltal az oktatók könnyen és strukturált módon rögzíthetik a vizsgákat az alkalmazásban, és biztosíthatják a szükséges információk egyszerű elérését a későbbi felhasználás céljából. Miután az új vizsga hozzá lett adva a rendszerhez, megjelenik egy táblázat, amelyben a hallgatók neptun kódjai soronként, az egyes feladatok azonosítói pedig oszloponként láthatóak. Ez az elrendezés lehetővé teszi a vizsgafeladatok és hallgatói válaszok strukturált és könnyen áttekinthető módon történő kezelését. A táblázatban minden cellában egy javítás gomb található, amelyre kattintva megjelenik a hallgató által írt feladat. Ez lehetővé teszi a javító oktátónak, hogy értékelje és pontozza a hallgató válaszát. Amennyiben egy feladat kijavításra került, a pontszám válik láthatóvá a javítás gomb helyett, így az oktatók könnyen nyomon követhetik a teljesítményt és az értékelést. Az összes vizsga kijavítását követően pedig megjelenik egy statisztika gomb, melyre kattintva megtekinthetjük az elért pontszámok átlagát, illetve az átlag pontszámot feladatonként.

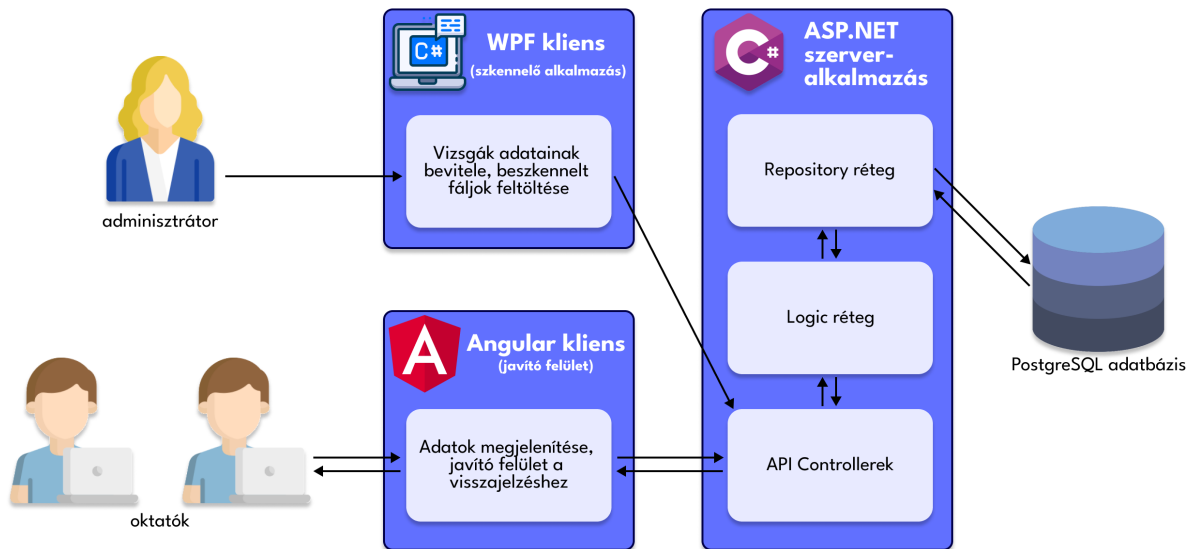
4.2.2 Funkcionális követelmények

- Bejelentkezési lehetőség az Office 365 fiókkal.
- Adminisztrátori jogok kezelése.
- Új vizsgák hozzáadása a rendszerhez.
- Hallgatói válaszok és pontok kezelése strukturált táblázatban.

4.3 Részletes rendszerterv

Ebben a fejezetben az alkalmazások belső működését szeretném bemutatni, hogy átfogóbb képet nyújtsak a rendszer működéséről.

A szkennelő alkalmazás API kontrollerek segítségével feltölti az adatbázisba a meglévő, feldolgozott adatokat az adott vizsgáról, minden logikai megvalósítás a repository rétegben található, a logic rétegben pedig ezen műveletek helyes működésének ellenőrzése kerül implementálásra. A javító alkalmazás továbbítja a felhasználó által küldött kéréseket a megfelelő végpont felé. Az alkalmazások működésének megértését a 4.3. ábra segíti, amely felvázolja a komponensek közötti kapcsolatokat.



4.2. ábra: Architektúrális ábra

4.3.1 Angular kliens és serveroldali ASP.NET alkalmazás kommunikációja

- **Felhasználók és szerepkörök kezelése:** A javító alkalmazásban az oktatók a saját Office 365 fiókjukkal tudnak bejelentkezni. Ezt az autentikációs folyamatot az ASP.NET Identity rendszer kezeli, amely biztosítja, hogy a felhasználók csak a megfelelő szerepkörhöz tartozó jogosultságokkal érhék el a rendszer egyes részeit. Az oktatók a bejelentkezés után kapják meg a hozzáférést a vizsgákhoz és a vizsgafolyamathoz, amelyhez az adminisztrátorok rendelik őket. Ezen funkciók elérhetősége a jogosultságkezelésen alapszik, amely az ASP.NET Identity-ben van beállítva. A serveroldal biztosítja, hogy a megfelelő felhasználók csak a hozzájuk rendelt vizsgákhoz férjenek hozzá.
- **Adatlekérdezés és megjelenítés a javító felületen:** A vizsgák adatainak lekérdezése az Angular alkalmazásból történik, ahol az oktatók megtekinthetik a hallgatók vizsgáit és válaszait. Az Angular kliens HTTP kéréseket küld az ASP.NET API kontrollereken keresztül, ahol a serveroldali alkalmazás logikája, valamint a repository réteg kapcsolódik

a PostgreSQL adatbázishoz, ahonnan a vizsgák adatai lekérhetőek. Az adatbázisban tárolt vizsgák és feladatok strukturált módon kerülnek vissza az Angular klienshez. A felületen ezek az adatok jelennek meg, lehetőséget biztosítva az oktatóknak a feladatok részletes megtekintésére és pontozására. Az oktatók a feladatokhoz fűzhetnek megjegyzéseket is, amelyek szintén API-n keresztül kerülnek feldolgozásra és mentésre az adatbázisba.

- **Pontozás és visszajelzés küldése:** A javító felület egyik legfontosabb funkciója a hallgatói válaszok értékelése. Az oktatók pontozzák a feladatokat, és a pontszámok mentése az ASP.NET alkalmazásban történik. Amikor az oktató beírja a pontszámokat, az Angular kliens API hívást küld a szervernek, amely feldolgozza az adatokat, és elmenti azokat a megfelelő hallgatói vizsgához. A pontozás során minden feladathoz külön rögzítésre kerül a hallgató válasza, a hozzá tartozó pontszám, és opcionálisan az oktatói visszajelzés. Ezek az adatok a PostgreSQL adatbázisban tárolódnak, és a szerveroldal biztosítja az adatok megfelelő feldolgozását és hitelességét.
- **Statisztikák generálása:** A javító felület egy másik fontos funkciója a vizsgaeredmények összegzése és statisztikák megjelenítése. Miután minden hallgató válaszát kijavították, az Angular kliens lehetőséget biztosít az eredmények összegzésére. Az API végpontokkal kommunikálva az alkalmazás a szerveren elvégzi az adatok összesítését, például a pontszámok átlagának kiszámítását vagy a statisztikai adatok feldolgozását. Ezek az adatok a szerveroldalon generált statisztikai jelentések formájában kerülnek vissza az Angular klienshez, ahol az oktatók áttekinthetik a hallgatók teljesítményét, és megtekinthetik a vizsgaeredmények alakulását.
- **Biztonság és adatvédelem:** Az adatbiztonság kiemelt fontosságú, mivel a rendszer személyes adatokat, például a hallgatók Neptun kódjait és vizsgafeladataikat kezeli. A szerveroldali ASP.NET alkalmazás biztosítja, hogy az adatok titkosítva és megfelelően védetten kerüljenek tárolásra a PostgreSQL adatbázisban. Emellett a rendszer gondoskodik arról, hogy csak az arra jogosult felhasználók férjenek hozzá az adatokhoz.
- **Adatfeldolgozás és mentés:** Az Angular kliens által küldött adatok a szerveroldali alkalmazás logikai rétegében kerülnek feldolgozásra. A logika biztosítja az adatok integritását, majd továbbítja azokat a repository réteg felé, amely az adatbázis műveleteket kezeli. A PostgreSQL adatbázisban történik az adatok végleges mentése, és a szerver felelős az adatok visszaadásáért a kliens felé.

4.3.2 Összegzés

Az architektúrális ábra alapján a javító alkalmazás és a szerveroldali ASP.NET alkalmazás közötti kommunikáció zökkenőmentesen történik a REST API végpontok segítségével. A szerveroldalon az adatok feldolgozása és tárolása biztonságos és jól strukturált, amely biztosítja a hallgatói válaszok és pontozás megbízható kezelését. Az Angular kliens szerepe az adatok lekérdezése, megjelenítése és a felhasználói interakciók továbbítása a szerveroldali logika felé.

5 MEGVALÓSÍTÁS

A megvalósítás során három fő komponens került kidolgozásra: a WPF szkennelő alkalmazás, a szerveroldali alkalmazás és a javító alkalmazás. Mindegyik komponens más-más szerepet tölt be a rendszer működésében, szorosan együttműködve biztosítják az adatok feldolgozását, mentését és megjelenítését.

5.1 WPF alkalmazás

A vizsgák feltöltési és feldolgozási folyamatának első lépéseként a felhasználónak be kell illesztenie az alkalmazásba a beszkennelt feladatsorokat tartalmazó mappát. Ezt követően meg kell adnia a vizsga alapvető adatait, például a tárgy nevét, a vizsga azonosítóját, a felelős oktató nevét és Neptun kódját, valamint a vizsga dátumát. A felhasználó értesítést kap arról, hogy minden szükséges adatot meg kell adnia a továbblépéshez. A felületet az 5.1. ábra szemlélteti:

Válassza ki vagy húzza be a képeket tartalmazó mappát!











5.1. ábra: Adatok megadása

5.1.1 Adatok feldolgozása

A megfelelő mappa beillesztése után az alkalmazás megkezdí a képek feldolgozását. Első lépésként végighalad a képeken, és amennyiben szükséges, automatikusan kiegyenesíti őket, biztosítva ezzel a hatékony feldolgozást. Ezután kivágja a releváns részeket – a Neptun kódokat és a feladat azonosítókat tartalmazó területeket –, és javítja a kivágott képrészletek minőségét. A feldolgozott adatokat egy Flask API-n keresztül továbbítja a TrOCR modellhez, amely karakterfelismerést végez, majd egy dictionary formátumú adatstruktúrában eltároljuk a kapott eredményt. Ez a dictionary a képek azonosítóit kulcsként, a Neptun kódokat és a feladatazonosítókat pedig értéként tartalmazza, így egyértelműen megjelöli, hogy melyik Neptun kód és feladatazonosító melyik képre vonatkozik.

Az adatok feldolgozási folyamatának következő lépése az automatikusan felismert adatok felhasználó általi ellenőrzése és szükség szerinti javítása. A felhasználónak lehetősége nyílik megtekinteni, hogy a program melyik képkockához milyen Neptun kódot és feladatazonosítót rendelt hozzá, és ha eltéréseket talál, manuálisan módosíthatja azokat.

A Neptun kódok véglegesítése előtt az alkalmazás a program által felismert kódokat összehasonlítja a felhasználó által előzetesen megadott listával. Az összehasonlítást a Levenshtein-távolság kiszámítására írt algoritmus segítségével végzi el, ami lehetővé teszi, hogy az automatikus felismerésből eredő kisebb eltéréseket pontosítsa. Az algoritmus kiszámítja a különbségeket a karakterláncok között, így a rendszer képes megtalálni az eredeti Neptun kódhoz legközelebbi változatot, és kijavítani az esetleges OCR-hibákat, azonban a ha az OCR hibásan felismer egy kódot, és ez épp közelebb esik egy másik helyes kódhoz a Levenshtein-távolság alapján, az hibás hozzárendeléshez vezethet. Amennyiben a TrOCR nem tud karaktersorozatot megállapítani, úgy visszatér a kép elérési útjára. Mindezen hibák kiküszöbölésére manuális ellenőrzést szükséges végrehajtanunk.

A folyamatot az 5.2. ábra szemlélteti, ahol a felhasználó láthatja a kódok összehasonlításának és javításának lépéseit. Ez az ellenőrzési és javítási lehetőség biztosítja az adatok helyességét, mielőtt a szerver felé való továbbításra kerülne sor.

The screenshot displays a web interface for data verification. At the top, there are two input fields for task codes: the first contains 'JT1PA7' and the second contains 'TZA962'. Below these, a text label reads 'ellezze az alábbi működést idő'. To the right, there are two forms. The first form, labeled 'Oktató adatai', contains fields for 'Kovács András' and 'ABC123', with 'Szerkesztés' and 'Mentés' buttons. The second form, labeled 'Vizsga adatai', contains fields for 'Szoftvertechnológia és grafikus felhasználói interfész', 'TZA960', and a date field '2024. 10. 25.', also with 'Szerkesztés' and 'Mentés' buttons. Below the input fields, there are two large text areas. The left one contains a list of task codes: LDE124, HIEEAB, JT1PA7, croppedNeptunCodes/img12_NeptunCode.png, TQ0PXY, LDE124, B4JXC1, K1XVVD, ASD987, KI3KSR, HIEEAB, LDE124, EIBK8V, and AJGMWS. The right one contains a list of task codes: TZA962, TZA964, TZA962, TZA964, TZA962, TZA964, TZA963, TZA963, TZA963, TZA963, TZA962, TZA963, TZA963, and TZA963. At the bottom, there are two buttons: 'croppedNeptunCodes/i' and 'TZA962', and a circular arrow icon.

5.2. ábra: Adatok ellenőrzése

5.1.2 Adatok továbbítása a szerver felé

Az adatok továbbítása a szerver felé a TaskUploader osztály UploadTaskSolutionsAsync metódusával történik, amely HTTP POST kérést küld egy JSON formátumú TaskSolutionDto objektummal a szerverre. Az adatok szerkezete magába foglalja a vizsga alapvető információit, a feladatok azonosítóit, a hallgatók megoldásait, pontszámait és a felelős oktató adatait. Ez a rendszer biztosítja az adatok strukturált továbbítását a szerver felé.

- Adatok előkészítése: A TaskSolutionDto objektum tartalmazza a vizsga azonosítóját, dátumát, tárgy nevét, oktató adatait és a feladatok listáját (TaskDto objektumok formájában). A TaskDto objektumok tárolják a feladat azonosítóját, a megoldásokat bináris formában (képek), a hallgatói azonosítókat, a pontszámokat és a maximális pontszámokat.
- Képfájlok beolvasása: Az egyes feladatok képfájljai bináris adatként kerülnek az objektumba. Ez biztosítja, hogy a szerverhez küldött adatok magukban foglalják a vizsgamegoldások digitális változatát.
- Kérés küldése: Az adatok küldése a HttpClient osztály segítségével történik, JSON formátumban. A JSON-objektumot a JsonConvert.SerializeObject metódus hozza létre, majd a PostAsync metódus küldi el a szerver /Exam/upload-data végpontjára.

5.2 Szerveroldali és Javító alkalmazás

5.2.1 Szerveroldali feldolgozás

A szerveroldali alkalmazás első lépésként ellenőrzi a beérkező adatok érvényességét, csak ezt követően kezdi meg az adatok feltöltését az adatbázisba.

- Vizsga létezésének ellenőrzése és hozzáadása: A metódus először megkeresi az adatbázisban az Exam táblában az ExamID alapján a vizsgát. Ha nem találja, egy új Exam objektum jön létre, amely tartalmazza a vizsga alapvető adatait, például az azonosítót, a tárgy nevét és a vizsga dátumát. Ezt követően az új objektumot hozzáadja az adatbázishoz, majd menti a módosításokat.
- Oktató létezésének ellenőrzése és hozzáadása: A metódus az Instructor tábla alapján megvizsgálja, hogy az oktató létezik-e már a Neptun kód alapján. Ha nem található az adott oktató, egy új Instructor objektumot hoz létre az oktató nevével és Neptun-kódjával, majd hozzáadja az adatbázishoz és menti a változtatásokat. A későbbiekben, az oktató regisztrációjakor az adatai automatikusan összekapcsolódnak a meglévő adatbázisrekorddal.
- Beadott megoldások feltöltése: Minden TaskDto objektumhoz létrehozunk egy Tasks objektumot, amely tartalmazza a feladat azonosítóját, a megoldást bináris formátumban, a Neptun kódot, a pontszámot és a maximális pontszámot. Ezeket az objektumokat a meglévő Exam objektumhoz adjuk hozzá. A vizsga összpontszámát a szkennelő alkalmazásban megadott feladatok pontszámainak összegére frissítjük.

5.2.2 Angular kliens

Az Angular kliens felhasználói felületet biztosít az oktatók számára a vizsgaadatok ellenőrzésére, javítására, visszajelzések küldésére és az eredmények nyomonkövetésére. A rendszer komponensalapú megközelítést alkalmaz, amely lehetővé teszi a nagy, összetett alkalmazások könnyű kezelhetőségét és karbantarthatóságát. **A projekt során az Angular következő jellemzőit és könyvtárait használtam:**

- Komponensalapú struktúra: Az alkalmazás minden funkciója külön komponensben van implementálva, például LoginComponent, ExamListComponent, ExamDetailsCompo-

nent.

- Routing és navigáció: Az Angular Routing modullal oldottam meg az alkalmazás oldalai közötti navigációt. A felhasználó autentikáció alapján kerül átirányításra az egyes oldalakra.
- FormsModule és ReactiveFormsModule: A bejelentkezési és regisztrációs űrlapok kezelésére.
- HttpClientModule: A szerverrel történő kommunikációhoz, például adatok lekérdezéséhez és küldéséhez.
- ToastrModule: Felhasználói értesítések (pl. sikeres mentés vagy hibüzenetek) megjelenítéséhez.
- Angular routing: Az alkalmazás több nézetének kezelésére és az oldalak közötti navigáció megvalósítására.
- NgClass és NgIf: Feltételes megjelenítés és stílus alkalmazása a felületen.
- Pipes: Adatok formázására, például dátumok és szövegek megjelenítéséhez.

Főbb funkciók és megvalósításuk:

1. Bejelentkezés és regisztráció

Funkcionalitás:

A kliens támogatja az oktatók bejelentkezését és új fiókok létrehozását. Az autentikáció során a szerver validálja a felhasználói adatokat, majd token-alapú azonosítást alkalmaz.

Megvalósítás:

A LoginComponent és RegisterComponent komponensek felelősek a bejelentkezési és regisztrációs oldalakért. Az Angular Reactive Forms modult használtam az űrlapok validációjához. A validációk magukban foglalják:

- Minimum karakterszámot.
- Formátumellenőrzést (például email cím esetén).
- Egyedi szabályokat, például a Neptun kód ellenőrzését regex segítségével.

2. Vizsgalistázás és szűrés A kliens egyik alapvető funkciója a vizsgák listázása és szűrése különböző paraméterek alapján (tantárgy, dátum, oktató). Ez lehetővé teszi, hogy az oktatók gyorsan megtalálják a releváns vizsgákat.

A vizsgaadatokat a szerverről HTTP GET kéréssel tölti be az alkalmazás. A szűrési opciók

ngModel kötésekkel kerülnek be az alkalmazás állapotába, amely alapján a megjelenített vizsgák listája automatikusan frissül. A rendezés a vizsgák attribútumai alapján valósul meg, például a dátum vagy pontszám növekvő/csökkenő sorrendjében.

3. Pontszámok ellenőrzése és javítása

A javító alkalmazás felületén az oktatók manuálisan ellenőrizhetik és módosíthatják a hallgatói megoldásokhoz tartozó pontszámokat, valamint megjegyzéseket fűzhetnek a hallgatói munkákhoz.

A pontszámok ellenőrzése egy táblázatos nézetben történik, amely *ngFor ciklussal generálja a hallgatókhoz tartozó adatokat. A módosítások során ngModel kötések és eseménykezelők biztosítják a pontszámok helyes frissítését. A mentés gomb HTTP POST kéréssel továbbítja az adatokat a szerver felé és az adatok validálását követően elmenti a módosításokat.

4. Statisztikák készítése és adatok exportálása Az alkalmazás lehetőséget biztosít vizsgastatisztikák készítésére, mint például az átlagpontszámok, hibaszázalékok vagy teljesítményeloszlások megjelenítésére, valamint az oktatók letölthetik a vizsgaeredményeket Excel formátumban, amely további elemzésekhez vagy adminisztrációhoz hasznosítható. Azxlsx könyvtár segítségével az adatok Excel fájlba kerülnek exportálásra, a fájl letöltéséhez pedig a File-Saver könyvtárat használtam. Az adatok grafikus megjelenítéséhez Chart.js-t használtam, a statisztikák az adatok elemzését követően grafikonokon jelennek meg.

6. Keresési funkció A keresés a vizsgák és hallgatók gyors megtalálására szolgál. Az oktatók kereshetnek tantárgyak, vizsgák vagy Neptun kódok alapján. A keresés az adatok szűrését követően dinamikusan frissíti az oldalt.

A megvalósított rendszer három fő komponensének – a WPF szkennelő alkalmazásnak, a szerveroldali alkalmazásnak és az Angular kliensnek – szoros együttműködése lehetővé teszi a vizsgafeldolgozási folyamat teljes körű támogatását.

6 TESZTELÉS ÉS ELÉRT EREDMÉNYEK

A projekt utolsó lépése az elkészített alkalmazások funkcióinak tesztelése és az elért eredmények vizsgálata.

6.1 Tesztelési környezet

A tesztelést az alábbi hardver- és szoftverkörnyezetben végeztem el, amely lehetővé tette az OCR rendszer teljesítményének és pontosságának részletes elemzését különböző adathalmazokon. A hardver- és szoftverparaméterek pontos dokumentálása biztosítja az eredmények későbbi reprodukálhatóságát és az egyéb környezetekkel való összehasonlíthatóságát.

- Hardver: Intel i7 processzor, 16 GB RAM, 512 GB SSD, NVIDIA GeForce RTX 3060 GPU.
- Operációs rendszer: Windows 10 x64.
- Fejlesztési eszközök: Visual Studio 2022, .NET 6.0, NUnit tesztkeretrendszer.
- Könyvtárak és technológiák: Python alapú OCR implementáció Flask API-val, C# .NET integráció.

6.2 Adatok elemzése

Az elforgatás és a vágás stabilan működő funkciói nem kerülnek részletes bemutatásra a jelen szakaszban, mivel ezek a korábbi fejlesztési ciklusok során már validálásra kerültek. A tesztelés jelenleg az OCR fő képességeire koncentrált, például a felismerési pontosságra és az eltérő minőségű adathalmazok kezelésére. Ez lehetővé tette a tesztelési erőforrások hatékonyabb elosztását, azokra a területekre fókuszálva, amelyek további finomhangolást igényelhetnek.

A 6.1. táblázat röviden bemutatja be az egyes adathalmazokon elért eredményeket:

Feldolg. képek száma	Helyesen felismerések sz.	Pontosság (%)	Feldolgozási idő (pp:mm)	Feldolgozási idő/Kép (ms)
40	35	87,50%	0:24	0,60
70	61	87,14%	0:39	0,57
120	98	81,67%	1:12	0,60
204	169	82,84%	2:07	0,62
402	338	84,08%	3:09	0,47

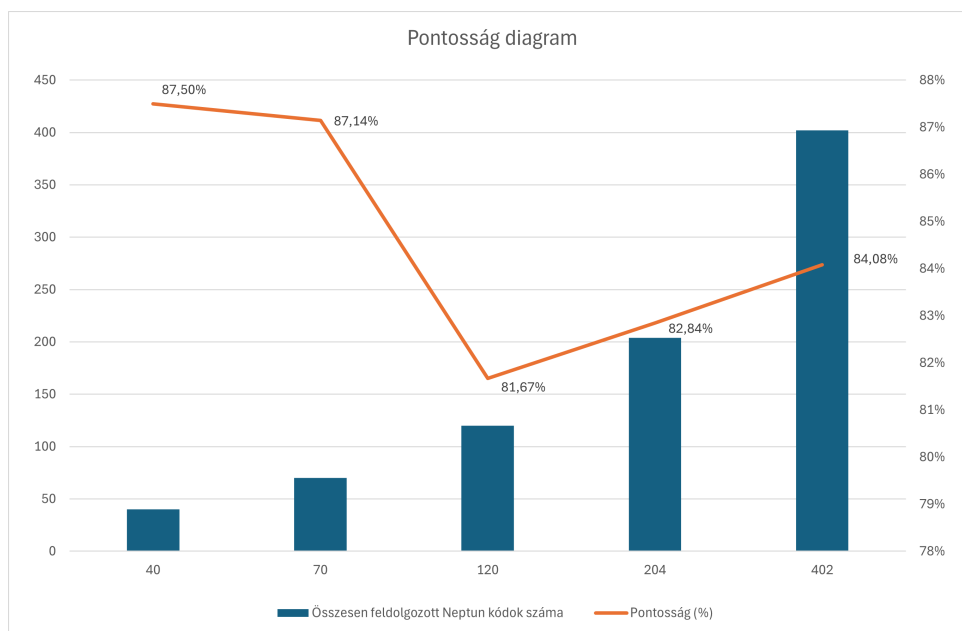
6.1. táblázat: Tesztelt adathalmazok és elért eredmények

A táblázatban látható, hogy a pontosság csökkenése a nagyobb adathalmazoknál valószínűleg az adatok változatosabb összetételének tudható be. Ez arra utal, hogy a rendszer érzékeny az adatok minőségére, amely a valós környezetben alkalmazott optimalizálási stratégiák fontos eleme lehet

6.2.1 Pontosság és hibaarány

Az OCR rendszer átlagosan 84,65%-os pontossággal dolgozott, amely azt mutatja, hogy a felismerések többsége helyes volt. Az átlagos hibaarány 15,35%, ami jelzi, hogy a rendszer bizonyos helyzetekben további finomítást igényelhet, például a kéziratos kódok nehezen olvasható esetekben történő felismerésében, valamint a megadott területen kívülre írt adatok terén. A tesztelt adathalmazokon belül a legjobb pontosságot (87,50%) a legkisebb méretű adathalmaz (40 kép) esetén, míg a legrosszabb pontosságot (81,67%) a 120 képet tartalmazó halmaznál érte el a rendszer. Ez az eredmény részben az adathalmazok eltérő minőségének és összetettségének tulajdonítható, például a képek tisztaságának vagy a kézírássok változékonyságának.

A 6.1.-es ábra az egyes adathalmazok pontosságának százalékos értékeit mutatja.

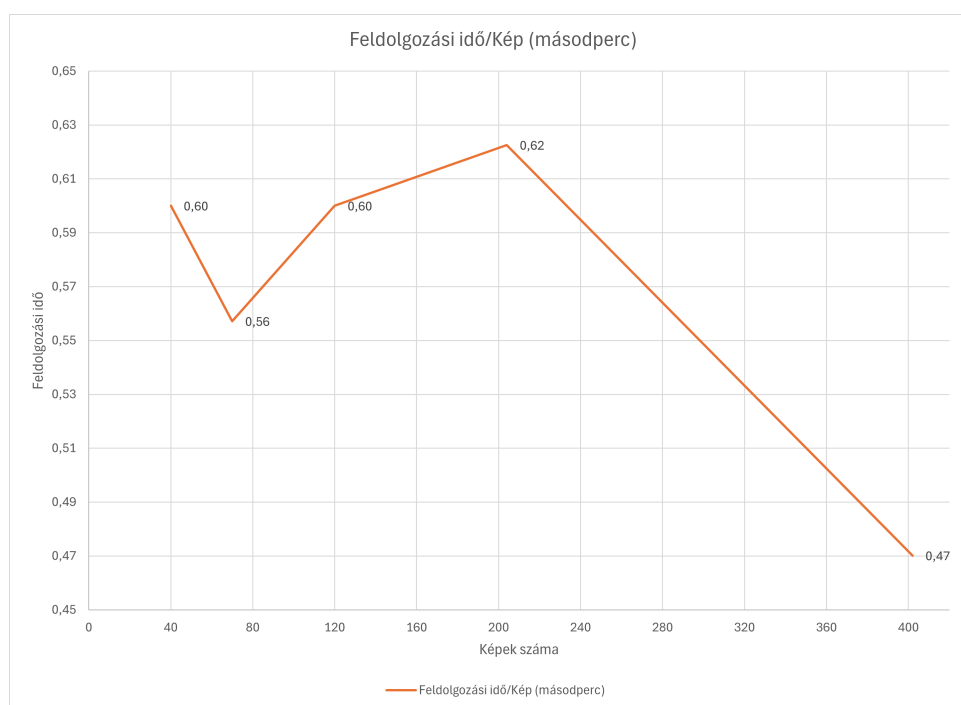


6.1. ábra: Pontosság a különböző adathalmazok esetén

6.2.2 Sebesség és feldolgozási idő

A feldolgozási sebesség szempontjából az OCR rendszer átlagosan 0,57 másodperc/kép alatt végezte el a felismerést, ami gyorsnak mondható kézírással adatok feldolgozásánál. A feldolgozási idő szórása mindössze 0,05 másodperc/kép, ami azt jelzi, hogy a rendszer teljesítménye stabil volt az egyes adathalmazokon belül. A nagyobb adathalmazok esetében a feldolgozási idő/kép csökkenése vagy szinten tartása arra utal, hogy a rendszer hatékonyan kezeli a párhuzamos feldolgozást, és jól alkalmazkodik a megnövekedett adatmennyiséghez. Ez fontos szempont lehet nagy mennyiségű adat feldolgozása során, ahol a skálázhatóság és a stabil teljesítmény kulcsfontosságú.

Az egyes adathalmazokra vonatkozó feldolgozási idő megoszlását az alábbi diagram mutatja:



6.2. ábra: Feldolgozási idő/kép másodpercben

6.2.3 Korlátok és relevancia

Bár az eredmények biztatóak, fontos hangsúlyozni, hogy az alkalmazott adathalmazok száma és jellege korlátozza a tesztelési környezet általános érvényességét:

- Adathalmazok száma: Mindössze öt adathalmaz állt rendelkezésre a teszteléshez, amelyek nem feltétlenül reprezentálják az OCR rendszer tényleges teljesítményét. További, eltérő

tulajdonságokkal rendelkező adathalmazok bevonása szükséges ahhoz, hogy átfogóbb képet kapjunk a rendszer teljesítményéről.

- Adathalmazok összetétele: Az egyes adathalmazok eltérő összetételűek voltak (például az azonosítók írásának minősége), ami részben hozzájárulhatott az eltérésekhez a pontosság és a feldolgozási idő terén.
- Adathalmazok homogenitása: A jelenlegi adathalmazok többnyire azonos típusú azonosítókat tartalmaztak, amelyek homogén jellegűek. Különböző formátumú és tartalmú karakterláncok tesztelése (például nevek vagy vegyesen számok és szövegek) szintén szükséges lehet a rendszer általános érvényességének biztosítására.

6.3 Eredmények összegzése

Az összesített statisztikákat a 6.2. táblázat szemlélteti:

Mutató	Érték
Átlagos pontosság (%)	84,65%
Pontosság szórása (%)	2,32%
Átlagos feldolgozási idő/kép (másodperc)	0,57
Feldolgozási idő/kép szórása (másodperc)	0,05
Összes feldolgozott kód	836
Összes helyesen felismert kód	701
Összes helytelenül felismert kód	135
Átlagos helyesen felismerési arány (%)	84,65%
Átlagos hibaarány (%)	15,35%
Teljes hibaarány az összes feldolgozott kódhoz (%)	16,15%
Legjobb adathalmaz	40 kép, pontosság: 87,50%
Legrosszabb adathalmaz	120 kép, pontosság: 81,67%
Adathalmaz: 40 kép, feldolgozási idő aránya (%)	5,32%
Adathalmaz: 70 kép, feldolgozási idő aránya (%)	8,65%
Adathalmaz: 120 kép, feldolgozási idő aránya (%)	15,96%
Adathalmaz: 204 kép, feldolgozási idő aránya (%)	28,16%
Adathalmaz: 402 kép, feldolgozási idő aránya (%)	41,91%

6.2. táblázat: A tesztelési eredmények összesített statisztikái

Konklúzió:

- Konzisztens teljesítmény: A rendszer feldolgozási ideje és pontossága viszonylag stabil az egyes adathalmazok között, ami azt mutatja, hogy megbízhatóan működik eltérő körülmények között is.
- Skálázhatóság: A nagyobb adathalmazoknál tapasztalt feldolgozási idő/kép csökkenése, illetve szinten tartása igazolja a rendszer hatékony párhuzamosítását, ami előnyt jelenthet valós környezetben, nagy mennyiségű adat feldolgozása esetén.
- Adathalmaz minőségének hatása: Az eredmények alapján az adathalmazok minősége és összetettsége jelentős szerepet játszik a pontosságban. Az elmosódott, rossz kontrasztú,

vagy a nem megfelelően pozícionált karakterek negatívan befolyásolták a felismerési pontosságot.

- Robosztusság: A rendszer a változó minőségű adathalmazok ellenére is elfogadható teljesítményt mutatott.

Az eredmények alapján az OCR rendszer stabil teljesítményt nyújt és képes volt az elvárt pontosságot és sebességet biztosítani, azonban a tesztelt adathalmazok homogenitása miatt további validálás szükséges eltérő környezetekben. Ez fontos a rendszer szélesebb körű alkalmazásához. A tesztelés eredményei biztatóak, de vannak olyan területek, ahol további fejlesztések segíthetik a rendszer teljesítményének javítását, különösen a kézírással írt karakterek felismerése és az adatok minőségének kezelésében. A rendszer jövőbeli alkalmazásaihoz hasznos lehet a tesztelés kiterjesztése többféle, különböző körülmények között készült képekre, ami segíthet a különböző alkalmazási területeken való szélesebb körű használatban. A finomhangolás és a további tesztelések révén az OCR rendszer további optimalizálására lesz lehetőség a jövőben.

7 TOVÁBBFEJLESZTÉSI LEHETŐSÉGEK

Az alkalmazás jelenlegi verziója már hatékonyan kezeli a vizsgák adatfeldolgozásának főbb kihívásait, ugyanakkor a rendszer további fejlesztésével jelentős mértékben növelhető a pontosság, a skálázhatóság és a felhasználói élmény. Az alábbiakban bemutatott javaslatok egymásra épülő, integrált módon valósíthatók meg, hogy a rendszer egységes és zökkenőmentes munkafolyamatokat biztosítson.

1. Automatikus és manuális folyamatok egyensúlyának megteremtése:

A manuális javítás integrálható a rendszer feldolgozási folyamataiba úgy, hogy az ne akadályozza a további feldolgozást. Ha az OCR bizonytalan egy kód azonosításában, vagy egyáltalán nem sikerül felismernie azt, akkor az érintett adatokat azonnal továbbíthatja a felhasználói ellenőrzési területre, miközben a többi adat feldolgozása folytatódik. Ez egy kézi validációs mód bevezetésével érhető el, ahol a felhasználók valós időben láthatják és javíthatják az OCR által bizonytalanul vagy hibásan felismert adatokat. A bizonytalan karaktereket az OCR karakteralapú bizalomértékei alapján kiemelve lehet megjeleníteni a felhasználók számára, hogy könnyebben azonosíthassák a problémás részeket.

2. Levenshtein-algoritmus és OCR bizalomértékek integrációja:

Az OCR által felismert karakterekhez rendelt bizalomértékek beépíthetők a Levenshtein-algoritmus működésébe, hogy pontosabb párosításokat lehessen elérni. Ez azt jelenti, hogy a Levenshtein-távolságot súlyozottan számítanánk, ahol a kevésbé biztos karakterek kisebb súllyal esnek latba az összehasonlítás során. Ez csökkenti annak valószínűségét, hogy egy tévesen felismert kód hibás párosítást eredményezzen. Az integráció részeként a rendszer képes lenne figyelembe venni a karakterek bizalomértékeit a párosítás során, és csak akkor továbbítaná a kódokat a listába, ha azok megfelelnek egy minimális bizonyossági küszöbértéknek.

3. Hibakezelési protokoll kiterjesztése és újrafeldolgozás:

Az OCR által nem felismert vagy bizonytalanul azonosított képeket automatikus újrafeldolgozási mechanizmusba lehetne helyezni. Az újrafeldolgozás során előfeldolgozási technikák, például további zajcsökkentés vagy kontrasztnövelés alkalmazásával javítható a képek minősége, majd a rendszer újra megpróbálhatja azok azonosítását. Ha a második próbálkozás sem jár sikerrel, a rendszer automatikusan jelentést generálhat az érintett adatokról, amelyeket a

manuális validációs folyamat során a felhasználók ellenőrizhetnek.

4. Batch-alapú feldolgozás és párhuzamosítás:

A rendszer skálázhatóságának és hatékonyságának növelése érdekében batch-alapú feldolgozást lehetne bevezetni. A feldolgozás során a beérkező képeket előre meghatározott méretű batch-ekre oszthatnánk, amelyeket a TrOCR modell párhuzamosan dolgozna fel. A batch-ek feldolgozásának eredményeit azonnal be lehetne tölteni az adatbázisba. Azon elemek, amelyeknél hibák merülnek fel, külön batch-be kerülhetnek, és automatikus újrafeldolgozás vagy manuális ellenőrzés alá vonhatóak.

5. Hallgatói felület kialakítása a tanulás elősegítéséhez

Egy hallgatói felület fejlesztése jelentős előnyökkel járhat a tanulás hatékonyságának növelésében. Az alkalmazás hallgatói felülete lehetőséget biztosíthatna arra, hogy a diákok hozzáférjenek az egyes tantárgyakhoz kapcsolódó vizsgafeladataikra kapott részletes visszajelzésekhez. Ezáltal gyorsan áttekinthetik az értékeléseket, valamint lehetőségük nyílik arra is, hogy jobban megértsék a tananyagra vonatkozó erősségeiket és gyengeségeiket és célzottan dolgozhatnak a fejlődésükön.

Emellett a hallgatói felületbe integrálható egy mesterséges intelligencia alapú chatbot, amely a feltöltött írásos tananyagokból automatikusan kérdéssorokat generálna. Ez lehetővé tenné a hallgatók számára, hogy a tanulás során rendszeresen gyakorolják a legfontosabb fogalmakat és kérdéseket, így a felkészülés sokkal interaktívabbá és dinamikusabbá válna. A chatbot nemcsak kérdéseket generálhat, hanem válaszokat is adhat a felmerülő kérdésekre, és képes lehet a hallgatók egyéni tanulási igényeihez alkalmazkodni.

6. Integrált hibakezelési és validációs rendszer:

Az összes fenti javaslatot egy egységes hibakezelési és validációs rendszerbe lehetne foglalni. Ez a rendszer valós időben kezelné a hibákat, biztosítaná a manuális és automatikus folyamatok integrációját, és lehetővé tenné a felhasználók számára, hogy az OCR által bizonytalanul felismert kódokat könnyedén ellenőrizzék. A rendszer egy átlátható jelentéskészítési modullal kiegészítve lehetőséget adna a fejlesztések hatékonyságának folyamatos nyomon követésére és a problémás területek azonosítására.

Ezek az integrált fejlesztési lépések nemcsak az alkalmazás teljesítményét és pontosságát

növelhetik, hanem jelentősen javíthatják a felhasználói élményt és a rendszer megbízhatóságát is, különösen nagyobb adathalmazok vagy komplexebb azonosítási feladatok esetén.

- Atawia, R., Hassanein, H. S., Abou-zeid, H., & Noureldin, A. (labeleearlabeleeardivisionlabel-monthlabelday). Robust Content Delivery and Uncertainty Tracking in Predictive Wireless Networks. *IEEE Transactions on Wireless Communications*, 16(4), 2327–2339. old. <https://doi.org/10.1109/TWC.2017.2662685>
- Atwood, S. A., & Singh, A. (labeleearlabeleeardivisionlabelmonthlabelday). Improved Pedagogy Enabled by Assessment Using Gradescope [<https://peer.asee.org/30627>]. *2018 ASEE Annual Conference & Exposition*, (10.18260/1-2–30627).
- Bao, H., Dong, L., Piao, S., & Wei, F. (labeleearlabeleeardivisionlabelmonthlabelday). BEiT: BERT Pre-Training of Image Transformers. <https://arxiv.org/abs/2106.08254>
- Canny, J. (labeleearlabeleeardivisionlabelmonthlabelday). A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6), 679–698. old. <https://doi.org/10.1109/TPAMI.1986.4767851>
- Cao, N., & Liu, Y. (labeleearlabeleeardivisionlabelmonthlabelday). High-Noise Grayscale Image Denoising Using an Improved Median Filter for the Adaptive Selection of a Threshold. *Applied Sciences*, 14(2). <https://doi.org/10.3390/app14020635>
- Developers, O. (labeleearlabeleeardivisionlabelmonthlabelday). Contour Features [Accessed: 2024-11-06].
- Documentation, O. (labeleearlabeleeardivisionlabelmonthlabelday). *Adding (Blending) Two Images Using OpenCV*. **retrieved** [urlyearurlyeardivisionurlmonthurlday](https://docs.opencv.org/4.x/d5/dc4/tutorial_adding_images.html)**from** https://docs.opencv.org/4.x/d5/dc4/tutorial_adding_images.html
- Edwards, L. (labeleearlabeleeardivisionlabelmonthlabelday). *What is Gradescope and How Can It Be Used to Teach? Tips & Tricks*. **retrieved** [urlyearurlyeardivisionurlmonthurlday](https://www.techlearning.com/how-to/what-is-gradescope-and-how-can-it-be-used-to-teach-tips-and-tricks)**from** <https://www.techlearning.com/how-to/what-is-gradescope-and-how-can-it-be-used-to-teach-tips-and-tricks>
- Jose, A., Deepa Merlin Dixon, K., Joseph, N., George, E. S., & Anjitha, V. (labeleearlabeleeardivisionlabelmonthlabelday). Performance study of edge detection operators. *2014 International Conference on Embedded Systems (ICES)*, 7–11. old. <https://doi.org/10.1109/EmbeddedSys.2014.6953040>

- Kanopoulos, N., Vasanthavada, N., & Baker, R. L. (labelyearlabelyeardivisionlabelmonthlabel-day). Design of an image edge detection filter using the Sobel operator. *IEEE Journal of solid-state circuits*, 23(2), 358–367. old.
- Li, M., Lv, T., Chen, J., Cui, L., Lu, Y., Florencio, D., Zhang, C., Li, Z., & Wei, F. (labelyearlabelyeardivisionlabelmonthlabel-day). TrOCR: Transformer-Based Optical Character Recognition with Pre-trained Models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(11), 13094–13102. old. <https://doi.org/10.1609/aaai.v37i11.26538>
- Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (labelyearlabelyeardivisionlabelmonthlabel-day). RoBERTa: A Robustly Optimized BERT Pretraining Approach. <https://arxiv.org/abs/1907.11692>
- Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (labelyearlabelyeardivisionlabelmonthlabel-day). You Only Look Once: Unified, Real-Time Object Detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 779–788. old. <https://doi.org/10.1109/CVPR.2016.91>
- Smith, R. (labelyearlabelyeardivisionlabelmonthlabel-day). An Overview of the Tesseract OCR Engine. *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, 2, 629–633. old. <https://doi.org/10.1109/ICDAR.2007.4376991>
- Surjono, H. D., PH, S., Priyanto & Suhendar. (labelyearlabelyeardivisionlabelmonthlabel-day). *The Effectiveness of the Zipgrade-Assisted Learning Outcomes Assessment Analysis in Promoting Indonesian Vocational Teachers' Competence*. **retrieved** [urlyearurlyeardivisionurlmonthurlday](https://eprints.untirta.ac.id/15552/2/c6%20Zipgrade.pdf) <https://eprints.untirta.ac.id/15552/2/c6%20Zipgrade.pdf>
- The Explorer, A. (labelyearlabelyeardivisionlabelmonthlabel-day). *How to Use ZIPGRADE? A Tutorial Guide for Teachers*. **retrieved** [urlyearurlyeardivisionurlmonthurlday](https://angelothexplorer.com/2019/08/how-to-use-zipgrade.html/) **from** <https://angelothexplorer.com/2019/08/how-to-use-zipgrade.html/>
- ZipGrade. (labelyearlabelyeardivisionlabelmonthlabel-day). *ZipGrade*. **retrieved** [urlyearurlyeardivisionurlmonthurlday](https://www.zipgrade.com/) <https://www.zipgrade.com/>

9 ÁBRAJEGYZÉK

2.1. Gradescope alkalmazás Atwood és Singh, labelyearlabelyeardivisionlabelmonth-labelday	7
2.2. Gradescope teszt létrehozása Atwood és Singh, labelyearlabelyeardivisionlabel-monthlabelday	8
2.3. Zipgrade alkalmazás	9
2.4. Saját osztály létrehozása	10
3.1. Feladatok sablonjai	12
3.2.	14
3.3. A Sobel-operátor alkalmazásának eredménye	15
3.4. Canny éldetektáló algoritmus végeredménye	16
3.5.	17
3.6.	18
3.7. Felismerés folyamata	21
4.1. Táblák és kapcsolatok	26
4.2. Architektúrális ábra	28
5.1. Adatok megadása	31
5.2. Adatok ellenőrzése	33
6.1. Pontosság a különböző adathalmazok esetén	38
6.2. Feldolgozási idő/kép másodpercben	39

10 RÖVIDÍTÉSEK

API	Application Programming Interface
NMS	Non-Maximum Suppression
OCR	Optical Character Recognition
YOLO	You Only Look Once
IOU	Intersection over Union
TrOCR	Transformer-based Optical Character Recognition
BEiT	BERT Pretrained Image Transformer
RoBERTa	Robustly Optimized BERT Approach
IAM	Institute of Advanced Motorists
CER	Character Error Rate