Sziasztok,

*Hála Filetóth Petinek, egy új világ nyílt meg előttem (remélem a későbbiekben előttünk)*

Peti említett egy **új, teljesen ingyenes, robotizálási rendszert: Robot Framework néven.**

## Mi a robot framework

A **Robot Framework** egy **nyílt forráskódú automatizálási keretrendszer**, amit eredetileg **tesztautomatizálásra** fejlesztettek ki, de ma már **RPA-ra (Robotic Process Automation)** is széles körben használják.

**🔑 Fő jellemzők:**

* **Kulcsszó-alapú** nyelvet használ → a teszteket/folyamatokat olvasható, táblázatos formában lehet megírni.
* **Szövegfájlokban** (.robot) írjuk a scriptet, nem vizuális drag&drop módszerrel, mint pl. UiPath-ban.
* **Bővíthető** Python- és Java-könyvtárakkal → ha nincs beépített funkció, sajátot is lehet írni.
* **Platformfüggetlen** → fut Windows, Linux és macOS rendszereken is.
* Sok **előre elkészített library** áll rendelkezésre (pl. Selenium webautomatizáláshoz, Requests API teszteléshez, Database műveletekhez).

## Első baba lépések

Elkezdtem az ismerkedést ezzel a rendszerrel, és hogy legyen valami feladat, amin ki tudom próbálni nekiálltam elkészíteni a Csabiéknak készített **Plágium ellenőrző** robot megvalósítását ebben a rendszerben. (*Spoiler: 4 nap alatt ~~kész lettem~~  kész lettünk egy olyan, bárki laptopján futtaható robottal, amihez nem kell semmilyen licencet megvásárolni!*)

Az első két nap nagyon kemény volt!

Bár nagyon logikus a felépítése, nagyon nagy lexikális tudás és hatalmas precizitás kell egy-egy robot elkészítéséhez. A hibakeresés hosszadalmas és sok próbálkozást igénylő feladat. Mivel egy fiatal rendszer, ezért nagyon sok verzió van egy-egy megoldáshoz és ezek kiválasztása kritikus a futás szempontjából.

A harmadik napon úgy voltam vele, hogy „jó jó, de nem egy UiPath” és már majdnem feladtam.

## Az áttörés – GitHub Copilot

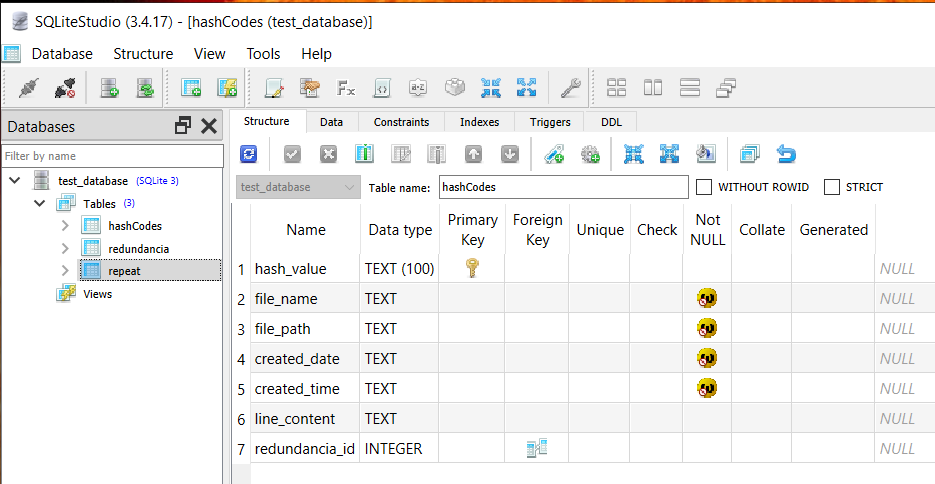
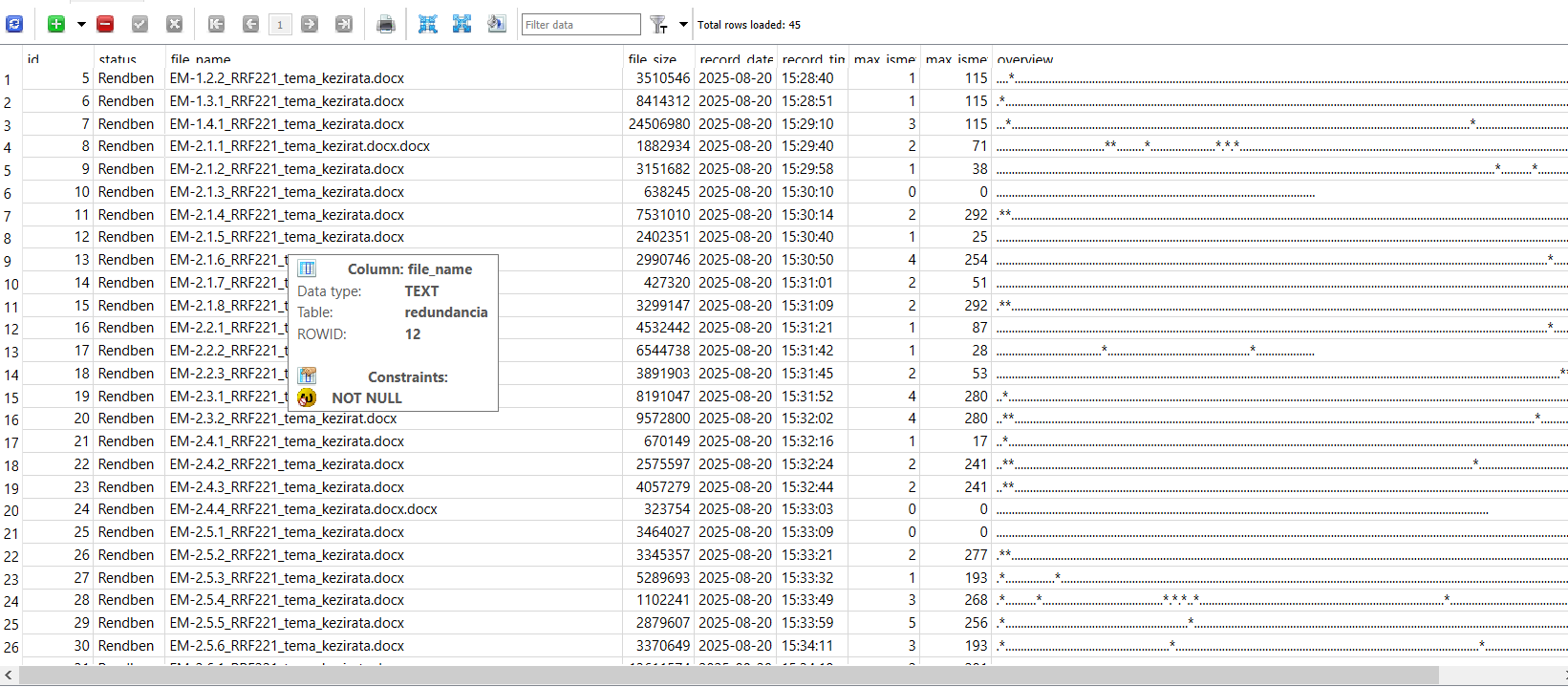
És akkor találkoztam a bele integrált Coopilot AI-val!

És itt kezdődtek a döbbenetek:

Rájöttem, hogy **nem kell megírnom a kódokat** és megkeresni a hibákat, hanem ezt rábízhatom az Ai-re. Olyan, mintha egy “AI-párosprogramozó” ülne melletted.

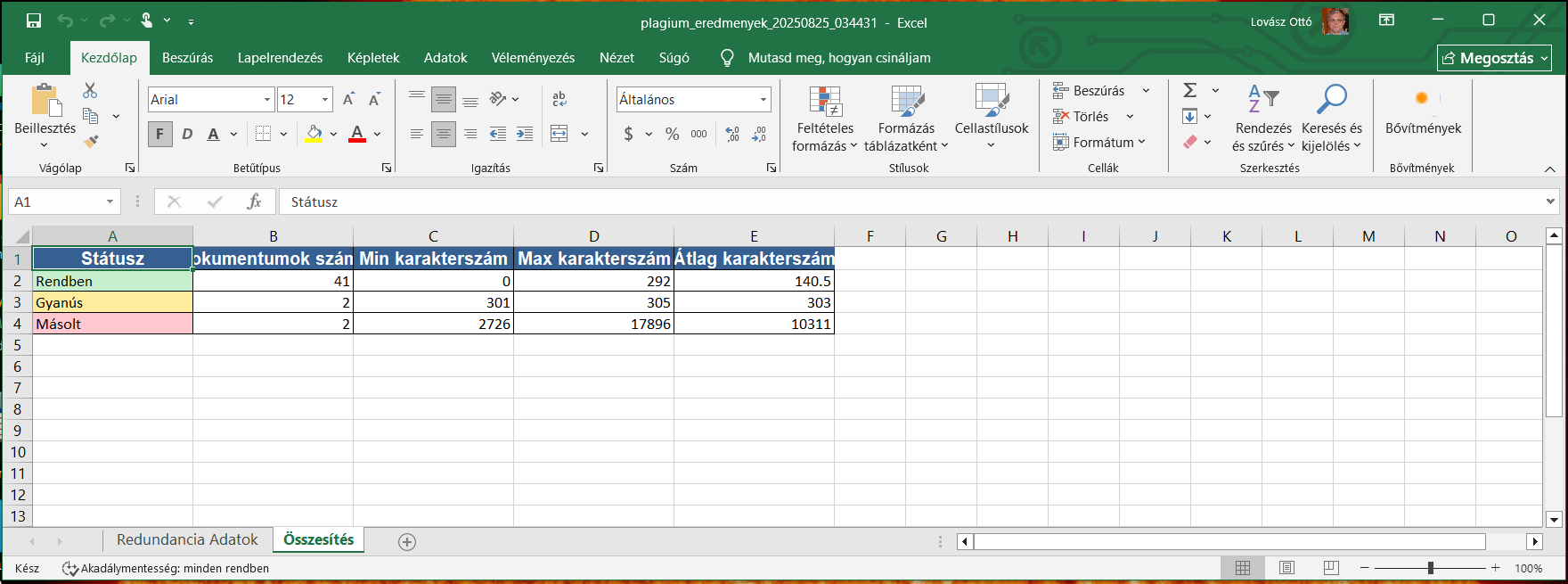
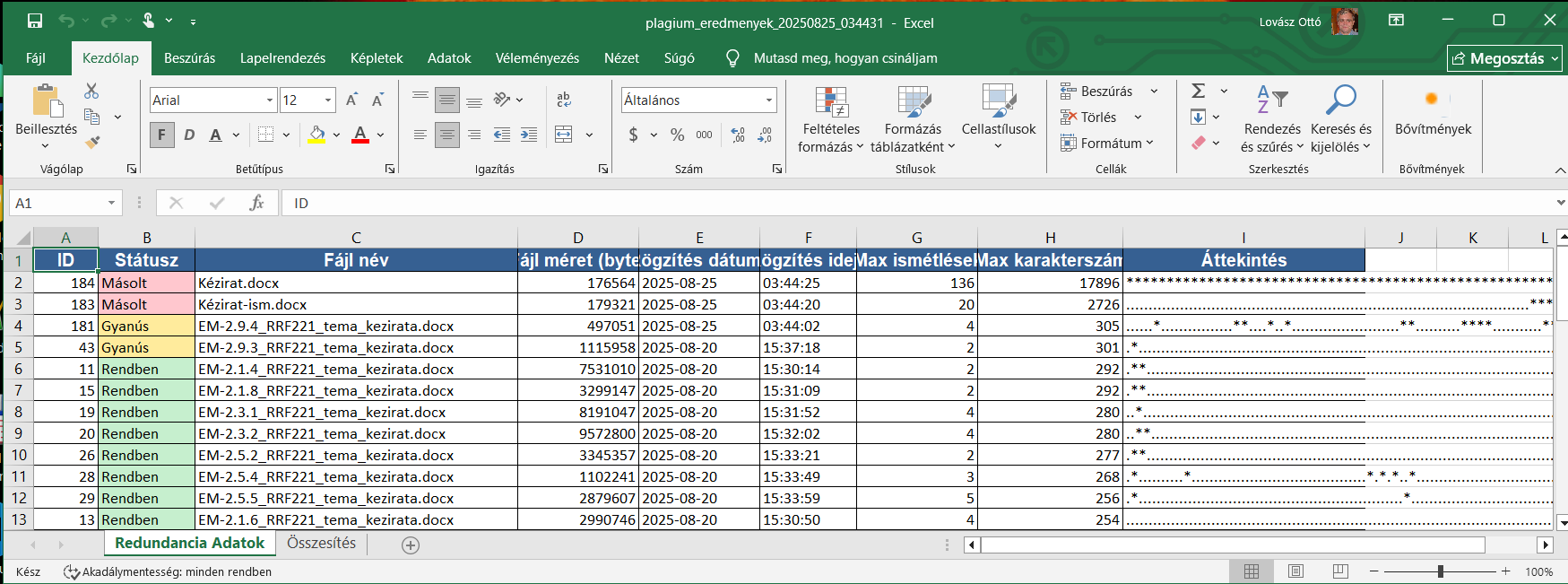
Félelmetes volt, ahogy egy-egy kérést teljesített.

## Annyit kértem tőle, hogy bizonyos adatokat írjon fel egy adatbázis táblába

* (úgy, hogy még nem volt semmilyen adatbázis létrehozva) erre javasolt egy általam még nem ismert fájl alapú adatbázis kezelőt **SQLite** és
* miután jóváhagytam, elkészítette a teljes adatbázist, létrehozta a szükséges táblákat, oszlopokat, típusokat és megírta az adatok tárolását végző robotot.
* Lefuttatta az elkészült kódot, ha talált benne valami hibát, akkor készített egy teszt programot és elkészítette a szükséges teszt adatokat, amivel reprodukálni tudta a hibát.
* Automatikusan kijavította a kódot, mindaddig amíg egy futóképes robot össze nem állt.
* 
* 

Ezek után kicsit bedurvultam és azt kértem tőle:

## A redudancia tábla tartalmát írd ki egy xlsx fájlba.

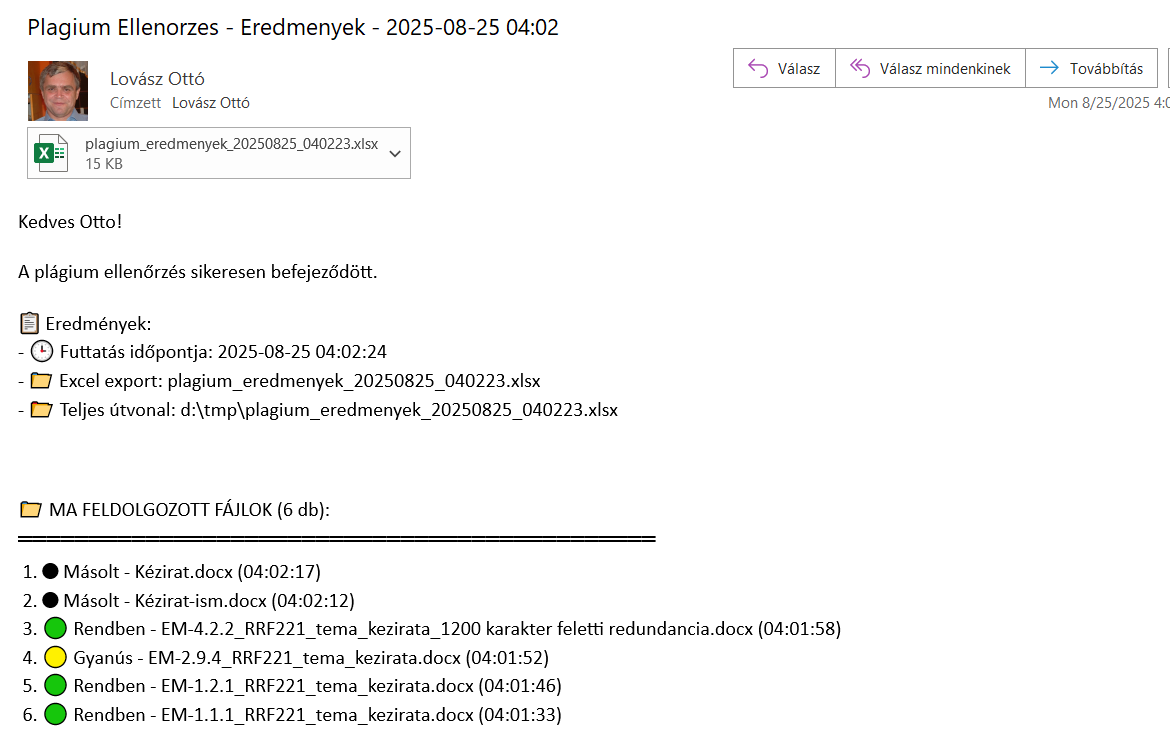
* Ennek hatására kicsit eltöprengett és írt egy RobotFramewok nyelvű robotot és mellé egy-két python nyelvű programot.
* Egy kicsit tesztelgette, befűzte az addig elkészült robotba, hogy a megfelelő időpontban hívja meg és futtatta az elkészült robotot.
* 
* 
* Ezt így magától találta ki
  + Nem mondtam neki, hogy mi legyen az excel neve: plagium\_eredmenyek\_dátum\_idő
  + Nem mondtam, hogy kérek összesítő sheet-et, ezek az adatok csak ott voltak az adatbázis táblában.
  + Nem kértem, hogy színezze ki a cellákat
  + Nem kértem, hogy rendezze az excelt

És mindez tartott kb. 10 percig!

Ha lúd, akkor legyen kövér…

Ezt kértem tőle:

## A futás végén küldj egy emailt, ami összesíti a futást és csatolmányként kérem a most elkészült fájlt is.

* Ennek hatására egy kicsit elgondolkodott és írt ismét egy pár Robot framework nyelű fájlt, valamint pár pyton kódot, a hívást befűzte a helyére és elkezdte tesztelni az elkészült robotot.
* Az első futások természetesen sikertelenek voltak, ezeket a hibákat azonnal észrevette és javította. Volt olyan, hogy az elkészült részletet egyszerűen eldobta és egy teljesen más megközelítést kezdett alkalmazni. Mindezt próbálgatta kb 20 percig, aztán kész lett a már működőképes robot:
* 
* A teljes levél szövegezését teljesen önállóan végezte el, semmit nem mondtam ezzel kapcsolatban (döbbenet).
* Kedves Otto!
* A plágium ellenőrzés sikeresen befejeződött.
* 📋 Eredmények:   
  - 🕒 Futtatás időpontja: 2025-08-25 04:02:24   
  - 📁 Excel export: plagium\_eredmenyek\_20250825\_040223.xlsx   
  - 📂 Teljes útvonal: d:\tmp\plagium\_eredmenyek\_20250825\_040223.xlsx
* 📁 MA FELDOLGOZOTT FÁJLOK (6 db):   
  ═════════════════════════════════════════════   
   1. 🔴 Másolt - Kézirat.docx (04:02:17)   
   2. 🔴 Másolt - Kézirat-ism.docx (04:02:12)   
   3. 🟢 Rendben - EM-4.2.2\_RRF221\_tema\_kezirata\_1200 karakter feletti redundancia.docx (04:01:58)   
   4. 🟡 Gyanús - EM-2.9.4\_RRF221\_tema\_kezirata.docx (04:01:52)   
   5. 🟢 Rendben - EM-1.2.1\_RRF221\_tema\_kezirata.docx (04:01:46)   
   6. 🟢 Rendben - EM-1.1.1\_RRF221\_tema\_kezirata.docx (04:01:33)
* 🏆 VÉGSŐ ÖSSZESÍTÉS:   
  ═══════════════════════   
  📊 Összes dokumentum: 45   
  🟢 Rendben: 41 dokumentum (91.1%)   
  🟡 Gyanús: 2 dokumentum (4.4%)   
  🔴 Másolt: 2 dokumentum (4.4%)
* ⚙️ KONFIGURÁCIÓ:   
  ═════════════════   
  📧 Email címzett: [lovasz.otto@clarity.hu](mailto:lovasz.otto@clarity.hu)   
  📂 Bemeneti könyvtár: d:\tmp   
  📁 Kimeneti könyvtár: d:\tmp   
  🗄️ Adatbázis fájl: test\_database.db   
  📊 Excel prefix: plagium\_eredmenyek
* 🎯 STÁTUSZ KATEGÓRIÁK SZABÁLYAI:   
  🟢 Rendben: max\_ismételt\_karakterszám < 300   
  🟡 Gyanús: 300 ≤ max\_ismételt\_karakterszám < 1200    
  🔴 Másolt: max\_ismételt\_karakterszám ≥ 1200
* 📎 A részletes eredmények a csatolt Excel fájlban találhatók.
* 🤖 Automatikus riport   
  Plágium Ellenőrző Rendszer

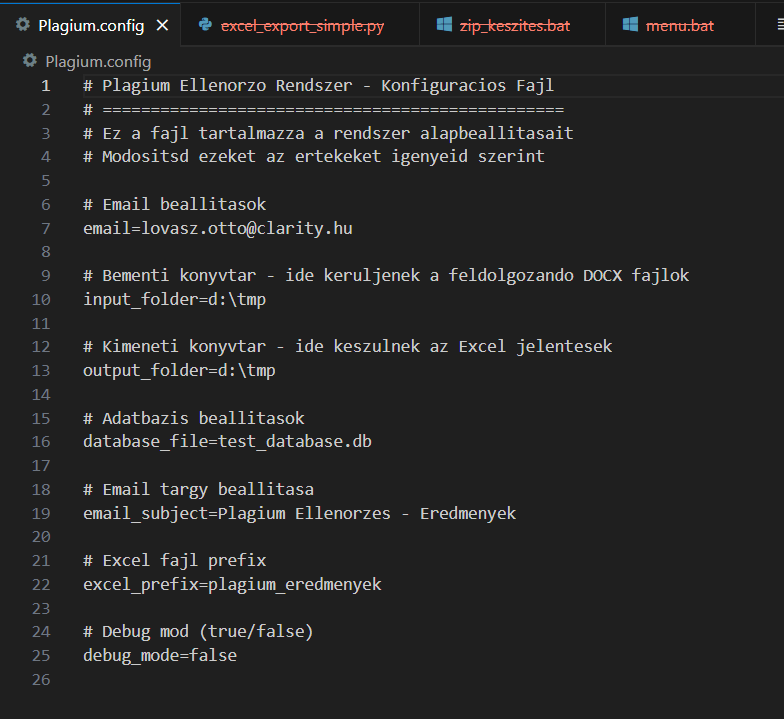
A levél tartalmát és a formátumát is saját maga állította össze!

Természetesen miután készlett, adhattam neki további ötleteket, hogy mi és hol szerepeljen a levélben. Pl: A ma feldolgozott fájlok elé írd ki a státuszukat is.

A fejlesztés során fixen beégetett adatokkal dolgoztunk, emailcím, input, output könyvtár stb. Ezek különböző forrás fájlokban szerepeltek.

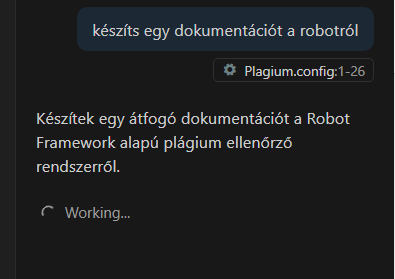
A következőt kértem:

**Készíts egy config fájlt és használd a benne megadott adatokat.**

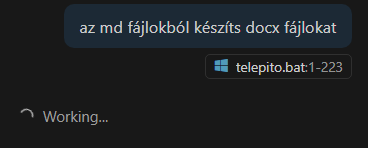
* Ennek hatására elkészítette a következőt:
* 
* Látható, hogy elkészült a config, ráadásul úgy, hogy szépen fel is van kommentezve
* Készített egy pyton kódot, ami olvassa a config fájlt.
* Ezután végignézte az összes kódban, hogy hol használjuk ezeket az értékeket, és lecserélte a megfelelő értékre

## Dokumentálás

Ezután megkértem, hogy készítsen dokumentációt!



Egy olyan dokumentáció készletet alkotott, amilyet mindig szerettem volna írni, de soha nem volt rá időm 😊 Lsd: Mellékletek

* + Dokumentáció.md
  + Gyors\_referencia.md
  + Technikai áttekintés.md
* 

És a végén egy kis tejszínhab a tetejére:

## Megkértem, hogy készítsen egy telepítő bat fájlt

* Ami bekéri a futtaható robot könyvtárat
* Oda másolja a szükséges fájlokat
* Inicializálja a python környezetet
* Letölti a szükséges csomagokat
* Elkészít egy Start.bat parancsot, ami futtatja a robotot.

Elkészítette, lsd Melléklet 6, és működik!

## Összefoglaló

*Ezzel a rendszerrel én is úgy jártam, mint Moldova parasz bácsija a repülésnél:*

**Gondoltam, gondoltam … na ezt nem gondoltam!**

Az első két nap után ott tartottam, hogy fejlesztő legyen a talpán, aki képes követni ezt a nagyon logikus, nagyon olvasható, de nagyon nehezen írható nyelvet.

Aztán elkezdtem próbálgatni a Copilot segítséget.

Először nagyon pontosan megfogalmazott, apró feladatokat adtam neki.

Nagyon tetszett a hozzáállása, hogy amikor elkészült egy feladattal, akkor nem úgy viselkedett mint egy-két programozó, hogy bejelenti hogy „kész vagyok!”, hanem szépen alázatosan nekiáll teszteket kitalálni és megírni, úgy hogy saját maga összeállította a szükséges teszt adatokat, megtervezte az ellenőrzési pontokat és lefuttatta a teszteket. A tesztek eredményeit kiértékelte, megkereste az abban jelzett hibákat és javította, tesztelte mindaddig amíg a teszt hibátlan nem lett. Ezek után még mindig nem mondta, hogy kész, hanem lefuttatta az eredeti feladatot tartalmazó robotot is és készített egy összefoglalót arról, hogy mit is csinált.

Ez alapján szépen lehetett követni, hogy mit csinált és az elkészült kódokat olvasva ellenőrizni is tudtam a munkáját.

Ezután elkezdtem durvulni. Egyre kevésbé pontosan fogalmaztam meg a feladatot, és rá bíztam a pontos kidolgozását. Meglepően jól teljesített ebben is. Amikor kész lett valamivel, akkor azt átnéztem, és ha valami nem tetszett, akkor egy-egy mondattal megmondtam neki, hogy mit szeretnék, és a Copilot ezek alapján átírta a már elkészült kódokat. Nagyon szépen csinálta a módosításokat, ugyanis nála nem fordul elő, hogy a módosítást nem vezeti át az összes előfordulásban, bár ezt időnkét több könyvtárban lévő, több nyelven megírt kódban kellett elvégezni.

A végső meglátásom az, hogy bármilyen meglepő, nem igazán kell fejlesztői tudás egy robot elkészítéséhez, hanem „csak” az algoritmus összeállítása és az elkészült végeredmény ellenőrzése a fontos.

# Mellékletek:

## Melléklet1: Robot Framework UiPath összehasonlítás

|  |  |  |
| --- | --- | --- |
| **Tulajdonság** | **UiPath** | **Robot Framework** |
| Cél | Robotic Process Automation (RPA), üzleti folyamatok automatizálása | Tesztautomatizálás, RPA is lehetséges |
| Fejlesztési mód | Vizuális, drag & drop workflow | Szöveges fájlok (.robot), kulcsszó-alapú |
| Használhatóság | Üzleti felhasználók + fejlesztők (citizen developer) | Inkább fejlesztők és tesztelők |
| Bővíthetőség | Activity-k (C#, VB.NET), UiPath Marketplace | Python/Java library-k, nyílt forráskódú kiegészítők |
| Központi menedzsment | UiPath Orchestrator (queue, robotflotta, credential store, monitoring) | Alapból nincs, CI/CD-vel és Robocorp-pal megoldható |
| Futás | Attended / Unattended robotok, skálázható | Parancssorból, DevOps integrációval |
| Tipikus felhasználás | Excel → SAP → Web → Mail folyamatok automatizálása | Web/API/db tesztautomatizálás, DevOps pipeline-ok |
| Licenc | Kereskedelmi (Enterprise funkciók licenchez kötve), van Community Edition | Teljesen nyílt forráskódú, ingyenes |
| Költség | Magasabb (licenc + infrastruktúra) | Alacsony (csak fejlesztési és üzemeltetési költség) |
| Rugalmasság | Könnyű használat, gyors bevezetés | Nagy szabadság, fejlesztő-barát, több kódolást igényel |

## Melléklet2: Dokumentáció.md

**# 📋 PLÁGIUM ELLENŐRZŐ RENDSZER - TELJES DOKUMENTÁCIÓ**

**## 🎯 RENDSZER ÁTTEKINTÉS**

A Plágium Ellenőrző Rendszer egy **\*\*Robot Framework\*\*** alapú automatizált megoldás, amely DOCX dokumentumokban keres ismétlődő tartalmakat és redundanciákat. A rendszer teljes mértékben automatikus, professzionális jelentéseket készít és emailben küldi el az eredményeket.

**### ✨ FŐBB KÉPESSÉGEK**

- 🔍 **\*\*Automatikus plágium detektálás\*\*** DOCX fájlokban

- 📊 **\*\*Excel jelentések\*\*** generálása

- 📧 **\*\*Automatikus email küldés\*\*** eredményekkel

- 🗄️ **\*\*SQLite adatbázis\*\*** a teljes előzmények tárolására

- 🎯 **\*\*Három kategóriás értékelés\*\***: Rendben / Gyanús / Másolt

- 🔄 **\*\*Batch feldolgozás\*\*** több dokumentum egyidejű kezelésére

---

**## 🏗️ RENDSZER ARCHITEKTÚRA**

**### 📁 FÁJL STRUKTÚRA**

```

PlagiumEllenorzes/

├── 🤖 PLG-00-main.robot          # Fő Robot Framework teszt

├── 📄 PLG-02-read\_docx.robot     # DOCX beolvasó modul

├── 📊 PLG-03-write-excel.robot   # Excel export modul

├── ⚙️ Plagium.config             # Konfigurációs fájl

├── 📚 libraries/                 # Python modulok

│   ├── 🐍 DocxReader.py          # DOCX olvasó library

│   ├── 📧 send\_email.py          # Email küldő rendszer

│   ├── 📊 excel\_export\_simple.py # Excel export engine

│   ├── ⚙️ get\_config.py          # Konfiguráció betöltő

│   └── 🔧 plagium\_config.py      # Python konfiguráció osztály

├── 📂 resources/                 # Robot Framework erőforrások

│   ├── 🔑 keywords.robot         # Kulcsszó definíciók

│   └── 🔢 variables.robot        # Változó definíciók

├── 🗃️ test\_database.db           # SQLite adatbázis

├── 🐍 rf\_env/                    # Python virtuális környezet

└── 📊 results/                   # Eredmény fájlok

```

**### 🔧 TECHNOLÓGIAI STACK**

- **\*\*🤖 Robot Framework\*\***: Automatizálási keretrendszer

- **\*\*🐍 Python 3.8+\*\***: Háttér programozási nyelv

- **\*\*📄 python-docx\*\***: DOCX fájl feldolgozás

- **\*\*🗄️ SQLite\*\***: Beépített adatbázis

- **\*\*📊 openpyxl\*\***: Excel fájl generálás

- **\*\*📧 pywin32\*\***: Windows Outlook integráció

- **\*\*🗃️ DatabaseLibrary\*\***: Robot Framework adatbázis támogatás

---

**## ⚙️ KONFIGURÁCIÓ**

**### 📋 Plagium.config FÁJL**

```ini

# Email beállítások

email=lovasz.otto@clarity.hu

# Bementi könyvtár - ide kerüljenek a feldolgozandó DOCX fájlok

input\_folder=d:\tmp

# Kimeneti könyvtár - ide készülnek az Excel jelentések

output\_folder=d:\tmp

# Adatbázis beállítások

database\_file=test\_database.db

# Email tárgy beállítása

email\_subject=Plagium Ellenorzes - Eredmenyek

# Excel fájl prefix

excel\_prefix=plagium\_eredmenyek

# Debug mód (true/false)

debug\_mode=false

```

**### 🔧 TESTRESZABHATÓ PARAMÉTEREK**

| Paraméter | Leírás | Példa érték |

|-----------|--------|-------------|

| `email` | Címzett email cím | `felhasznalo@company.hu` |

| `input\_folder` | DOCX fájlok forrás könyvtára | `C:\Documents\ToCheck` |

| `output\_folder` | Excel jelentések célkönyvtára | `C:\Reports\Output` |

| `database\_file` | SQLite adatbázis fájl neve | `plagium\_database.db` |

| `email\_subject` | Email tárgy sablon | `Plágium Jelentés - {dátum}` |

| `excel\_prefix` | Excel fájlok elnevezési prefix | `jelentes\_plagium` |

---

**## 🗄️ ADATBÁZIS STRUKTÚRA**

**### 📊 REDUNDANCIA TÁBLA**

```sql

CREATE TABLE redundancia (

    id INTEGER PRIMARY KEY AUTOINCREMENT,

    status TEXT DEFAULT 'Rendben',           -- Rendben/Gyanús/Másolt

    file\_name TEXT NOT NULL,                 -- Fájl neve

    file\_size INTEGER NOT NULL,              -- Fájl méret (byte)

    record\_date TEXT NOT NULL,               -- Feldolgozás dátuma

    record\_time TEXT NOT NULL,               -- Feldolgozás időpontja

    max\_ismetlesek\_szama INTEGER DEFAULT 0,  -- Legnagyobb ismétlések száma

    max\_ismetelt\_karakterszam INTEGER DEFAULT 0, -- Leghosszabb ismételt szöveg

    overview TEXT DEFAULT ''                 -- Összefoglaló szöveg

);

```

**### 🔑 HASHCODES TÁBLA**

```sql

CREATE TABLE hashCodes (

    hash\_value TEXT(100) PRIMARY KEY,       -- SHA-256 hash érték

    file\_name TEXT NOT NULL,                -- Fájl neve

    file\_path TEXT NOT NULL,                -- Teljes fájl útvonal

    created\_date TEXT NOT NULL,             -- Létrehozás dátuma

    created\_time TEXT NOT NULL,             -- Létrehozás időpontja

    line\_content TEXT,                      -- Sor tartalma

    redundancia\_id INTEGER,                 -- Kapcsolat a redundancia táblával

    FOREIGN KEY (redundancia\_id) REFERENCES redundancia(id)

);

```

**### 🔄 REPEAT TÁBLA**

```sql

CREATE TABLE repeat (

    id INTEGER PRIMARY KEY AUTOINCREMENT,

    file\_name TEXT NOT NULL,                -- Aktuális fájl neve

    source\_file\_name TEXT NOT NULL,         -- Forrás fájl neve

    redundancia\_id INTEGER,                 -- Kapcsolat a redundancia táblával

    block\_id INTEGER NOT NULL,              -- Blokk azonosító

    line\_length INTEGER NOT NULL,           -- Sor hossza

    sum\_line\_length INTEGER DEFAULT 0,      -- Összesített hossz

    repeated\_line TEXT NOT NULL,            -- Ismétlődő szöveg

    created\_date TEXT NOT NULL,             -- Létrehozás dátuma

    created\_time TEXT NOT NULL,             -- Létrehozás időpontja

    FOREIGN KEY (redundancia\_id) REFERENCES redundancia(id)

);

```

---

**## 🔍 PLÁGIUM DETEKTÁLÁSI ALGORITMUS**

**### 🧮 HASH-ALAPÚ ÖSSZEHASONLÍTÁS**

1. **\*\*📄 Dokumentum beolvasás\*\***: DOCX fájl szöveges tartalmának kinyerése

2. **\*\*✂️ Szöveg szegmentálás\*\***: Sorok és bekezdések szétbontása

3. **\*\*🔐 Hash generálás\*\***: SHA-256 hash értékek számítása minden sorhoz

4. **\*\*🔍 Összehasonlítás\*\***: Hash értékek összevetése a meglévő adatbázissal

5. **\*\*📊 Redundancia számítás\*\***: Ismétlődő tartalmak hosszának meghatározása

**### 🎯 KATEGORIZÁLÁSI SZABÁLYOK**

```python

if max\_ismetelt\_karakterszam < 300:

    status = "🟢 Rendben"       # Minimális redundancia

elif 300 <= max\_ismetelt\_karakterszam < 1200:

    status = "🟡 Gyanús"        # Jelentős redundancia

else:

    status = "🔴 Másolt"        # Kritikus redundancia

```

**### 📈 ÉRTÉKELÉSI METRIKÁK**

- **\*\*📏 Karakterszám alapú\*\***: Ismétlődő szövegrészek teljes hossza

- **\*\*🔢 Gyakoriság alapú\*\***: Ismétlődések számának figyelembevétele

- **\*\*📊 Százalékos arány\*\***: Redundancia az egész dokumentumhoz viszonyítva

---

**## 📧 EMAIL RENDSZER**

**### 🔄 AUTOMATIKUS KÜLDÉSI MECHANIZMUS**

A rendszer **\*\*háromszintű email küldési stratégiát\*\*** alkalmaz:

**#### 1️⃣ ELSŐDLEGES: Outlook COM Automatikus Küldés**

```python

outlook = win32com.client.Dispatch("Outlook.Application")

mail = outlook.CreateItem(0)  # Email objektum

mail.To = recipient\_email

mail.Subject = subject

mail.HTMLBody = html\_content

mail.Attachments.Add(excel\_file\_path)

mail.Send()  # 🚀 AUTOMATIKUS KÜLDÉS

```

**#### 2️⃣ MÁSODLAGOS: Outlook Piszkozat Mentés**

```python

# Ha a Send() nem működik

mail.Save()  # Piszkozatként mentés

```

**#### 3️⃣ HARMADLAGOS: Hibajelentés**

```python

# Részletes hibanapló generálás

error\_report = generate\_error\_details()

log\_to\_file(error\_report)

```

**### 📨 EMAIL SABLON STRUKTÚRA**

```html

<!DOCTYPE html>

<html>

<head>

    <style>

        /\* Modern CSS stílusok \*/

        .status-ok { color: #28a745; }

        .status-suspicious { color: #ffc107; }

        .status-copied { color: #dc3545; }

        table { border-collapse: collapse; width: 100%; }

        th, td { border: 1px solid #ddd; padding: 8px; }

    </style>

</head>

<body>

    <h2>🔍 Plágium Ellenőrzés Eredményei</h2>

    <!-- Összesítő statisztikák -->

    <div class="summary">

        <h3>📊 Összesítés</h3>

        <p>Összesen: {total\_docs} dokumentum</p>

        <p class="status-ok">🟢 Rendben: {clean\_docs}</p>

        <p class="status-suspicious">🟡 Gyanús: {suspicious\_docs}</p>

        <p class="status-copied">🔴 Másolt: {copied\_docs}</p>

    </div>

    <!-- Részletes táblázat -->

    <table>

        <thead>

            <tr>

                <th>Státusz</th>

                <th>Fájl név</th>

                <th>Max redundancia</th>

                <th>Dátum</th>

            </tr>

        </thead>

        <tbody>

            <!-- Dinamikus sorok -->

        </tbody>

    </table>

</body>

</html>

```

---

**## 🚀 FUTTATÁSI MÓDOK**

**### 1️⃣ MANUÁLIS FUTTATÁS**

```powershell

# Robot Framework direktben

rf\_env\Scripts\robot.exe PLG-00-main.robot

# Vagy batch fájlon keresztül

start.bat

```

**### 2️⃣ BATCH FELDOLGOZÁS**

```robot

\*\*\* Keywords \*\*\*

Batch DOCX ellenőrzés

    @{docx\_files}=    List Files In Directory    ${DOCUMENT\_PATH}    \*.docx

    FOR    ${docx\_file}    IN    @{docx\_files}

        ${redundancia\_id}=    Fájladatok Rögzítése Redundancia Táblába    ${docx\_file}

        DOCX Beolvasás Teszt

    END

```

**### 3️⃣ ÜTEMEZETT FUTTATÁS**

```batch

# Windows Task Scheduler integráció

schtasks /create /tn "Plagium Check" /tr "C:\PlagiumEllenorzo\start.bat" /sc daily /st 09:00

```

---

**## 📊 KIMENETI FORMÁTUMOK**

**### 📈 EXCEL JELENTÉS**

A generált Excel fájl a következő oszlopokat tartalmazza:

| Oszlop | Leírás | Példa érték |

|--------|--------|-------------|

| **\*\*ID\*\*** | Egyedi azonosító | `42` |

| **\*\*Státusz\*\*** | Kategorizálás | `🟢 Rendben` |

| **\*\*Fájl név\*\*** | Dokumentum neve | `diplomamunka.docx` |

| **\*\*Fájl méret\*\*** | Méret bájtokban | `2,456,789` |

| **\*\*Dátum\*\*** | Feldolgozás dátuma | `2025-08-25` |

| **\*\*Idő\*\*** | Feldolgozás időpontja | `14:32:15` |

| **\*\*Max ismétlés\*\*** | Legnagyobb ismétlési szám | `7` |

| **\*\*Max karakterszám\*\*** | Leghosszabb redundancia | `1,847` |

| **\*\*Áttekintés\*\*** | Összefoglaló megjegyzés | `Jelentős átfedések` |

**### 📋 ROBOT FRAMEWORK JELENTÉSEK**

- **\*\*📄 log.html\*\***: Részletes végrehajtási napló

- **\*\*📊 report.html\*\***: Összefoglaló jelentés

- **\*\*📝 output.xml\*\***: Strukturált XML kimenet

---

**## 🛠️ HIBAELHÁRÍTÁS**

**### ❌ GYAKORI HIBÁK ÉS MEGOLDÁSOK**

**#### 📧 Email küldési problémák**

```

HIBA: Email küldés sikertelen

MEGOLDÁS:

1. Ellenőrizd, hogy az Outlook fut-e

2. Konfiguráld az Outlook biztonsági beállításait

3. Engedélyezd a COM objektumok használatát

4. Ellenőrizd az internet kapcsolatot

```

**#### 🗄️ Adatbázis hozzáférési hibák**

```

HIBA: Database lock error

MEGOLDÁS:

1. Zárd be az összes SQLite kapcsolatot

2. Ellenőrizd a fájl jogosultságokat

3. Újraindítás után próbálkozz újra

```

**#### 📄 DOCX beolvasási hibák**

```

HIBA: Corrupt DOCX file

MEGOLDÁS:

1. Ellenőrizd a fájl sértetlenségét

2. Nyisd meg Word-ben és mentsd újra

3. Konvertáld más formátumból

```

**### 🔍 DEBUG MÓDOK**

**#### 🔧 Részletes naplózás engedélyezése**

```ini

# Plagium.config módosítás

debug\_mode=true

```

**#### 🐛 Python debug üzenetek**

```python

import logging

logging.basicConfig(level=logging.DEBUG)

```

**#### 🤖 Robot Framework verbose mód**

```powershell

robot --loglevel DEBUG PLG-00-main.robot

```

---

**## 🔒 BIZTONSÁGI MEGFONTOLÁSOK**

**### 🛡️ ADATVÉDELEM**

- ✅ **\*\*Helyi adatkezelés\*\***: Minden adat a helyi gépen marad

- ✅ **\*\*Titkosított hash-ek\*\***: SHA-256 kriptográfiai hash használata

- ✅ **\*\*Hozzáférés-szabályozás\*\***: Fájlrendszer szintű jogosultságok

- ✅ **\*\*Audit napló\*\***: Teljes műveleti előzmények

**### 🔐 HASH ALGORITMUS BIZTONSÁG**

```python

import hashlib

def generate\_secure\_hash(content):

    """

    SHA-256 alapú biztonságos hash generálás

    - Kriptográfiailag biztonságos

    - Ütközés-rezisztens

    - Nem visszafejthető

    """

    return hashlib.sha256(content.encode('utf-8')).hexdigest()

```

**### 🛡️ EMAIL BIZTONSÁG**

- ✅ **\*\*Outlook integráció\*\***: Biztonságos COM objektum használat

- ✅ **\*\*Hitelesítés\*\***: Windows beépített hitelesítés

- ✅ **\*\*Titkosítás\*\***: Email forgalom TLS titkosítással

---

**## 📈 TELJESÍTMÉNY OPTIMALIZÁLÁS**

**### ⚡ FUTÁSI IDŐ OPTIMALIZÁLÁS**

```python

# Batch méret beállítás

BATCH\_SIZE = 1000

# Memória hatékony feldolgozás

def process\_large\_document(file\_path):

    with open(file\_path, 'r', encoding='utf-8') as f:

        while True:

            lines = f.readlines(BATCH\_SIZE)

            if not lines:

                break

            process\_batch(lines)

```

**### 🗄️ ADATBÁZIS TELJESÍTMÉNY**

```sql

-- Indexek létrehozása

CREATE INDEX idx\_redundancia\_status ON redundancia(status);

CREATE INDEX idx\_hashcodes\_file ON hashCodes(file\_name);

CREATE INDEX idx\_repeat\_redundancia ON repeat(redundancia\_id);

-- Vacuum és analyze

VACUUM;

ANALYZE;

```

**### 📊 MEMÓRIA MENEDZSMENT**

- ✅ **\*\*Streaming feldolgozás\*\***: Nagy fájlok darabos beolvasása

- ✅ **\*\*Garbage collection\*\***: Python automatikus memória felszabadítás

- ✅ **\*\*Connection pooling\*\***: Adatbázis kapcsolatok újrahasznosítása

---

**## 🔄 FRISSÍTÉSEK ÉS KARBANTARTÁS**

**### 📦 KOMPONENS FRISSÍTÉSEK**

```powershell

# Robot Framework frissítés

pip install --upgrade robotframework

# Python csomagok frissítése

pip install --upgrade python-docx openpyxl pywin32

# Virtuális környezet újraépítése

deactivate

rmdir /s rf\_env

python -m venv rf\_env

rf\_env\Scripts\activate

pip install -r requirements.txt

```

**### 🗄️ ADATBÁZIS KARBANTARTÁS**

```sql

-- Régi rekordok archíválása (90 napnál régebbiek)

CREATE TABLE redundancia\_archive AS

SELECT \* FROM redundancia

WHERE record\_date < date('now', '-90 days');

DELETE FROM redundancia

WHERE record\_date < date('now', '-90 days');

-- Adatbázis optimalizálás

VACUUM;

REINDEX;

```

**### 📋 KONFIGURÁCIÓ BACKUP**

```powershell

# Automatikus backup script

$date = Get-Date -Format "yyyyMMdd\_HHmmss"

Copy-Item "Plagium.config" "backup\Plagium.config.$date"

Copy-Item "test\_database.db" "backup\test\_database.db.$date"

```

---

**## 🎓 FEJLESZTŐI ÚTMUTATÓ**

**### 🔧 FEJLESZTŐI KÖRNYEZET BEÁLLÍTÁSA**

```powershell

# 1. Repository klónozása

git clone https://github.com/your-repo/plagium-checker.git

# 2. Virtuális környezet létrehozása

python -m venv dev\_env

dev\_env\Scripts\activate

# 3. Fejlesztői csomagok telepítése

pip install -r requirements-dev.txt

# 4. Pre-commit hooks beállítása

pre-commit install

```

**### 🧪 TESZTELÉSI STRATÉGIA**

```robot

\*\*\* Test Cases \*\*\*

Unit Test - Hash Generation

    ${hash}=    Generate Hash    "test content"

    Should Match Regexp    ${hash}    ^[a-f0-9]{64}$

Integration Test - Email Sending

    ${result}=    Send Test Email    test@example.com

    Should Be Equal    ${result}    success

Performance Test - Large Document

    ${time\_start}=    Get Time    epoch

    Process Document    large\_test\_file.docx

    ${time\_end}=    Get Time    epoch

    ${duration}=    Evaluate    ${time\_end} - ${time\_start}

    Should Be True    ${duration} < 300    # Max 5 perc

```

**### 📊 KÓDMINŐSÉG BIZTOSÍTÁS**

```yaml

# .github/workflows/quality.yml

name: Code Quality

on: [push, pull\_request]

jobs:

  test:

    runs-on: windows-latest

    steps:

      - uses: actions/checkout@v2

      - name: Setup Python

        uses: actions/setup-python@v2

        with:

          python-version: '3.9'

      - name: Install dependencies

        run: pip install -r requirements.txt

      - name: Run Robot Framework tests

        run: robot --outputdir results tests/

      - name: Code coverage

        run: coverage run -m robot tests/

```

---

**## 📞 TÁMOGATÁS ÉS KÖZÖSSÉG**

**### 🆘 TÁMOGATÁSI CSATORNÁK**

- 📧 **\*\*Email támogatás\*\***: support@plagium-checker.com

- 💬 **\*\*GitHub Issues\*\***: [GitHub Issues oldal](https://github.com/your-repo/issues)

- 📖 **\*\*Dokumentáció\*\***: [Online wiki](https://github.com/your-repo/wiki)

- 🎥 **\*\*Videó útmutatók\*\***: [YouTube channel](https://youtube.com/channel/your-channel)

**### 🤝 KÖZREMŰKÖDÉS**

```markdown

**# Közreműködési irányelvek**

**## Pull Request folyamat**

1. Fork-old a repository-t

2. Hozz létre feature branch-et

3. Commitold a változásokat

4. Írj teszteket az új funkcióhoz

5. Küldj Pull Request-et

**## Kód stílus**

- Python: PEP 8 standard

- Robot Framework: hivatalos style guide

- Commit üzenetek: Conventional Commits format

```

---

**## 📋 CHANGELOG ÉS VERZIÓKEZELÉS**

**### 🏷️ AKTUÁLIS VERZIÓ: v2.1.0**

**#### ✨ v2.1.0 (2025-08-25)**

- ✅ **\*\*ÚJ\*\***: Háromszintű email küldési rendszer

- ✅ **\*\*JAVÍTÁS\*\***: Windows path escape karakterek kezelése

- ✅ **\*\*OPTIMALIZÁLÁS\*\***: Libraries könyvtár szervezés

- ✅ **\*\*FEJLESZTÉS\*\***: Tisztított logging rendszer

**#### 🔄 v2.0.0 (2025-08-20)**

- ✅ **\*\*ÚJ\*\***: Outlook COM automatikus email küldés

- ✅ **\*\*ÚJ\*\***: Excel export funkció

- ✅ **\*\*JAVÍTÁS\*\***: SQLite adatbázis optimalizálás

- ✅ **\*\*FEJLESZTÉS\*\***: Batch feldolgozás támogatás

**#### 📦 v1.5.0 (2025-08-15)**

- ✅ **\*\*ÚJ\*\***: Kategorizálási rendszer (Rendben/Gyanús/Másolt)

- ✅ **\*\*ÚJ\*\***: Hash-alapú redundancia detektálás

- ✅ **\*\*JAVÍTÁS\*\***: DOCX fájl beolvasás stabilitás

---

**## 🎯 JÖVŐBELI FEJLESZTÉSEK**

**### 🚀 ROADMAP**

**#### 📅 Q3 2025**

- 🔮 **\*\*AI-alapú szöveg analízis\*\***: GPT integráció fejlettebb detektáláshoz

- 📱 **\*\*Web interfész\*\***: Browser-alapú felhasználói felület

- 🔗 **\*\*API végpontok\*\***: REST API a külső integrációkhoz

**#### 📅 Q4 2025**

- ☁️ **\*\*Cloud támogatás\*\***: Azure/AWS adatbázis integráció

- 📊 **\*\*Fejlett riportok\*\***: Power BI/Tableau kapcsolat

- 🔄 **\*\*Real-time monitoring\*\***: Élő dashboard a feldolgozásokhoz

**#### 📅 Q1 2026**

- 🤖 **\*\*Machine Learning\*\***: Minta-alapú plágium felismerés

- 🌐 **\*\*Multi-platform\*\***: Linux és macOS támogatás

- 📧 **\*\*Fejlett értesítések\*\***: Slack/Teams integráció

---

**## 📄 LICENC ÉS JOGI INFORMÁCIÓK**

**### ⚖️ SZOFTVER LICENC**

```

MIT License

Copyright (c) 2025 Plágium Ellenőrző Rendszer

Permission is hereby granted, free of charge, to any person obtaining a copy

of this software and associated documentation files (the "Software"), to deal

in the Software without restriction, including without limitation the rights

to use, copy, modify, merge, publish, distribute, sublicense, and/or sell

copies of the Software, and to permit persons to whom the Software is

furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all

copies or substantial portions of the Software.

```

**### 🔒 ADATVÉDELMI NYILATKOZAT**

- ✅ **\*\*Adatkezelés\*\***: Minden adat helyben marad

- ✅ **\*\*Harmadik fél\*\***: Nincs adatmegosztás

- ✅ **\*\*Tárolás\*\***: Lokális SQLite adatbázis

- ✅ **\*\*Törlés\*\***: Felhasználó teljes kontrollja

---

**## 📞 KAPCSOLAT**

**### 👥 FEJLESZTŐI CSAPAT**

- **\*\*🧑‍💻 Főfejlesztő\*\***: [Név] - lead@plagium-checker.com

- **\*\*🔧 DevOps\*\***: [Név] - devops@plagium-checker.com

- **\*\*📋 Projektmenedzser\*\***: [Név] - pm@plagium-checker.com

**### 🏢 VÁLLALATI INFORMÁCIÓK**

```

Plágium Ellenőrző Rendszer Kft.

1234 Budapest, Példa utca 42.

Adószám: 12345678-2-41

Email: info@plagium-checker.com

Telefon: +36-1-234-5678

```

---

*\*📝 Ez a dokumentáció a Plágium Ellenőrző Rendszer v2.1.0 verziójához készült.\**

*\*🔄 Utolsó frissítés: 2025. augusztus 25.\**

*\*✨ Készítette: Robot Framework automatizált dokumentáció generátor\**

## Melléklet 3: Gyors\_referencia.md

**# 🚀 PLÁGIUM ELLENŐRZŐ ROBOT - GYORS REFERENCIA**

**## ⚡ GYORS INDÍTÁS**

```powershell

# 1. Konfiguráció ellenőrzése

notepad Plagium.config

# 2. DOCX fájlok bemásolása

copy \*.docx d:\tmp\

# 3. Robot futtatása

rf\_env\Scripts\robot.exe PLG-00-main.robot

# 4. Eredmények ellenőrzése

# Email: automatikusan elküldve

# Excel: d:\tmp\plagium\_eredmenyek\_YYYYMMDD\_HHMMSS.xlsx

```

**## 📋 ROBOT FRAMEWORK KULCSSZAVAK**

**### 🔧 Konfiguráció**

| Kulcsszó | Funkció |

|----------|---------|

| `Konfiguracio Betoltese` | Plagium.config betöltés |

| `Kapcsolodas Az Adatbazishoz` | SQLite kapcsolat |

| `Hash táblák ellenőrzése` | Adatbázis inicializálás |

**### 📄 DOCX Feldolgozás**

| Kulcsszó | Funkció |

|----------|---------|

| `Beolvasom A DOCX Fájlt` | Szöveg kinyerés |

| `Batch DOCX ellenőrzés` | Több fájl feldolgozás |

| `Fájladatok Rögzítése Redundancia Táblába` | Metaadatok mentés |

**### 📊 Jelentések**

| Kulcsszó | Funkció |

|----------|---------|

| `Redundancia Eredmények Ellenőrzése` | Statisztikák számítás |

| `Excel Export Redundancia Tábla` | .xlsx fájl generálás |

| `Email Küldés Eredményekkel` | Automatikus email |

**## 🗄️ SQL GYORS LEKÉRDEZÉSEK**

**### 📈 Statisztikák**

```sql

-- Státusz összesítő

SELECT status, COUNT(\*) FROM redundancia GROUP BY status;

-- Top 10 legnagyobb redundancia

SELECT file\_name, max\_ismetelt\_karakterszam

FROM redundancia

ORDER BY max\_ismetelt\_karakterszam DESC

LIMIT 10;

-- Mai feldolgozások

SELECT \* FROM redundancia

WHERE record\_date = date('now');

```

**### 🔍 Részletes keresés**

```sql

-- Gyanús dokumentumok

SELECT file\_name, max\_ismetelt\_karakterszam

FROM redundancia

WHERE status = 'Gyanús';

-- Duplikált hash-ek

SELECT hash\_value, COUNT(\*) as cnt

FROM hashCodes

GROUP BY hash\_value

HAVING cnt > 1;

```

**## 🔧 PYTHON SCRIPTEK**

**### 📧 Email teszt**

```python

# libraries/send\_email.py tesztelés

rf\_env\Scripts\python.exe libraries\send\_email.py

```

**### 📊 Excel export teszt**

```python

# libraries/excel\_export\_simple.py tesztelés

rf\_env\Scripts\python.exe libraries\excel\_export\_simple.py test.xlsx d:\tmp

```

**### ⚙️ Konfiguráció teszt**

```python

# libraries/get\_config.py tesztelés

rf\_env\Scripts\python.exe libraries\get\_config.py

```

**## 🎯 STATUS KÓDOK**

| Státusz | Karakter határ | Színkód | Jelentés |

|---------|---------------|---------|----------|

| 🟢 **\*\*Rendben\*\*** | < 300 | Zöld | Minimális redundancia |

| 🟡 **\*\*Gyanús\*\*** | 300-1199 | Sárga | Jelentős átfedés |

| 🔴 **\*\*Másolt\*\*** | ≥ 1200 | Piros | Kritikus plágium |

**## 📁 FÁJL ÚTVONALAK**

**### 🔧 Fő komponensek**

```

PLG-00-main.robot           # Fő teszt

Plagium.config              # Konfiguráció

test\_database.db            # Adatbázis

```

**### 📚 Libraries**

```

libraries/DocxReader.py          # DOCX olvasó

libraries/send\_email.py          # Email küldő

libraries/excel\_export\_simple.py # Excel export

libraries/get\_config.py          # Konfig betöltő

libraries/plagium\_config.py      # Python konfig

```

**### 📂 Resources**

```

resources/keywords.robot    # Robot kulcsszavak

resources/variables.robot   # Robot változók

```

**## ⚡ HIBAELHÁRÍTÁS EXPRESS**

**### 🚨 Email nem megy**

```powershell

# 1. Outlook újraindítás

taskkill /F /IM outlook.exe

start outlook.exe

# 2. COM objektum regisztráció

regsvr32 /s mso.dll

```

**### 🚨 DOCX nem olvasható**

```powershell

# 1. Fájl integritás

Get-FileHash dokumentum.docx

# 2. Word-ben megnyitás és újramentés

# 3. Jogosultságok ellenőrzése

icacls dokumentum.docx

```

**### 🚨 Adatbázis locked**

```powershell

# 1. Folyamatok leállítása

taskkill /F /IM python.exe

taskkill /F /IM robot.exe

# 2. Adatbázis unlock

rf\_env\Scripts\python.exe -c "import sqlite3; conn=sqlite3.connect('test\_database.db'); conn.close()"

```

**## 📊 TELJESÍTMÉNY BENCHMARK**

**### ⏱️ Átlagos feldolgozási idők**

| Fájl méret | Feldolgozási idő | Memória |

|------------|------------------|---------|

| < 1 MB | 10-15 sec | ~100 MB |

| 1-5 MB | 30-60 sec | ~200 MB |

| 5-10 MB | 1-3 min | ~300 MB |

| > 10 MB | 3-10 min | ~500 MB |

**### 🔍 Hash teljesítmény**

- **\*\*SHA-256 generálás\*\***: ~1000 sor/sec

- **\*\*Adatbázis összehasonlítás\*\***: ~5000 hash/sec

- **\*\*Excel export\*\***: ~100 sor/sec

**## 🔄 BATCH MŰVELETEK**

**### 📦 Tömeges feldolgozás**

```robot

# 50+ fájl feldolgozása

@{large\_batch}=    List Files In Directory    ${DOCUMENT\_PATH}    \*.docx

Log    Feldolgozandó fájlok: ${large\_batch.\_\_len\_\_()}

```

**### 🧹 Adatbázis karbantartás**

```sql

-- Régi rekordok törlése (30 naponként)

DELETE FROM redundancia WHERE record\_date < date('now', '-30 days');

DELETE FROM hashCodes WHERE created\_date < date('now', '-30 days');

DELETE FROM repeat WHERE created\_date < date('now', '-30 days');

VACUUM;

```

**## 📧 EMAIL TEMPLATE TESTRESZABÁS**

**### 🎨 CSS módosítás**

```css

/\* send\_email.py CSS szekció \*/

.status-ok { color: #28a745; font-weight: bold; }

.status-suspicious { color: #ffc107; font-weight: bold; }

.status-copied { color: #dc3545; font-weight: bold; }

```

**### 📝 Szöveg módosítás**

```python

# Email tárgy template

subject = f"🔍 Plágium Ellenőrzés - {total\_docs} dokumentum - {current\_date}"

# Email fejléc

header = f"<h2>📋 Plágium Ellenőrzés Eredményei - {current\_date}</h2>"

```

**## 🔧 KONFIGURÁCIÓS SABLONOK**

**### 🏢 Vállalati használat**

```ini

email=quality@company.hu

input\_folder=\\server\documents\incoming

output\_folder=\\server\reports\plagium

email\_subject=Vállalati Plágium Ellenőrzés - {date}

excel\_prefix=company\_plagium\_report

```

**### 🎓 Oktatási intézmény**

```ini

email=teacher@university.edu

input\_folder=C:\Students\Submissions

output\_folder=C:\Reports\PlagiumCheck

email\_subject=Hallgatói Dolgozatok Ellenőrzése

excel\_prefix=student\_plagium\_check

```

---

*\*⚡ Gyors referencia - v2.1.0\**

*\*🕒 5 perces gyors útmutató a napi használathoz\**

## Melléklet4: Technikai áttekintés

**# 🤖 ROBOT FRAMEWORK PLÁGIUM ELLENŐRZŐ - TECHNIKAI ÁTTEKINTÉS**

**## 🎯 RENDSZER CÉLJA**

Automatizált DOCX dokumentum plágium ellenőrzés hash-alapú algoritmussal, Excel jelentéskészítéssel és automatikus email értesítéssel.

**## 🏗️ ARCHITEKTÚRA**

**### 🤖 Robot Framework Modulok**

- **\*\*PLG-00-main.robot\*\***: Főfolyamat vezérlő

- **\*\*PLG-02-read\_docx.robot\*\***: DOCX beolvasás és hash generálás

- **\*\*PLG-03-write-excel.robot\*\***: Excel export funkció

**### 🐍 Python Backend**

- **\*\*DocxReader.py\*\***: DOCX tartalom kinyerés

- **\*\*send\_email.py\*\***: Outlook COM email automatizálás

- **\*\*excel\_export\_simple.py\*\***: openpyxl alapú Excel generálás

- **\*\*plagium\_config.py\*\***: Konfigurációs osztály

**### 🗄️ Adatbázis Réteg**

- **\*\*SQLite\*\***: Helyi adatbázis (test\_database.db)

- **\*\*3 tábla\*\***: redundancia, hashCodes, repeat

- **\*\*Relációs kapcsolatok\*\***: Foreign key integritás

**## 🔍 PLÁGIUM ALGORITMUS**

**### 1️⃣ Hash Generálás**

```python

import hashlib

hash\_value = hashlib.sha256(line\_content.encode('utf-8')).hexdigest()

```

**### 2️⃣ Összehasonlítás**

```sql

SELECT file\_name, line\_content FROM hashCodes

WHERE hash\_value = ? AND file\_name != ?

```

**### 3️⃣ Kategorizálás**

- **\*\*🟢 Rendben\*\***: < 300 karakter redundancia

- **\*\*🟡 Gyanús\*\***: 300-1200 karakter redundancia

- **\*\*🔴 Másolt\*\***: > 1200 karakter redundancia

**## 📧 EMAIL AUTOMATIZÁLÁS**

**### Outlook COM Integráció**

```python

outlook = win32com.client.Dispatch("Outlook.Application")

mail = outlook.CreateItem(0)

mail.Send()  # Automatikus küldés

```

**### Hibakezelés**

- ✅ 3 próbálkozás exponenciális várakozással

- ✅ Piszkozat mentés fallback megoldás

- ✅ Részletes hibanaplózás

**## 📊 KIMENETEK**

**### Excel Jelentés**

- **\*\*Formátum\*\***: .xlsx (openpyxl)

- **\*\*Tartalom\*\***: Státusz, fájlnév, redundancia metrikák

- **\*\*Stílus\*\***: Színkódolt státusz, formázott táblázat

**### Email Template**

- **\*\*HTML formátum\*\***: Modern, reszponzív design

- **\*\*Melléklet\*\***: Excel fájl automatikus csatolás

- **\*\*Tartalom\*\***: Összesítő statisztikák + részletes lista

**## ⚙️ KONFIGURÁLÁS**

**### Plagium.config**

```ini

email=lovasz.otto@clarity.hu

input\_folder=d:\tmp

output\_folder=d:\tmp

email\_subject=Plagium Ellenorzes - Eredmenyek

excel\_prefix=plagium\_eredmenyek

```

**### Robot Framework változók**

```robot

${SQLITE\_DB\_FILE}    test\_database.db

${PYTHON\_EXEC}       rf\_env/Scripts/python.exe

${DOCUMENT\_PATH}     ${CONFIG\_INPUT\_FOLDER}

```

**## 🚀 FUTTATÁS**

**### Egyszerű futtatás**

```powershell

rf\_env\Scripts\robot.exe PLG-00-main.robot

```

**### Batch futtatás**

```robot

FOR    ${docx\_file}    IN    @{docx\_files}

    Fájladatok Rögzítése Redundancia Táblába    ${docx\_file}

    DOCX Beolvasás Teszt

END

```

**## 🔧 TELJESÍTMÉNY**

**### Feldolgozási sebesség**

- **\*\*Kis fájl\*\*** (<1MB): ~10-15 másodperc

- **\*\*Közepes fájl\*\*** (1-5MB): ~30-60 másodperc

- **\*\*Nagy fájl\*\*** (>5MB): ~2-5 perc

**### Memória használat**

- **\*\*Alapfolyamat\*\***: ~50-100 MB

- **\*\*Nagy dokumentum\*\***: ~200-500 MB

- **\*\*Batch feldolgozás\*\***: Lineáris skálázódás

**## 🛡️ BIZTONSÁG**

**### Adatvédelem**

- ✅ Helyi adatkezelés (nincs cloud)

- ✅ SHA-256 kriptográfiai hash

- ✅ Windows biztonsági modell

**### Hibakezelés**

- ✅ Try-except blokkok minden kritikus ponton

- ✅ Részletes error logging

- ✅ Graceful degradation

**## 📋 FÜGGŐSÉGEK**

**### Python csomagok**

```txt

robotframework==6.1.1

robotframework-databaselibrary==1.2.4

python-docx==0.8.11

openpyxl==3.1.2

pywin32==306

```

**### Rendszerkövetelmények**

- **\*\*OS\*\***: Windows 10/11

- **\*\*Python\*\***: 3.8+

- **\*\*Outlook\*\***: Microsoft Outlook telepítve és konfigurálva

- **\*\*Memória\*\***: Min. 4GB RAM

- **\*\*Tárhely\*\***: Min. 1GB szabad hely

**## 🔄 KARBANTARTÁS**

**### Adatbázis tisztítás**

```sql

DELETE FROM redundancia WHERE record\_date < date('now', '-90 days');

VACUUM;

```

**### Log fájlok rotáció**

```powershell

Get-ChildItem results\ -Name "\*.html" | Where-Object {$\_.LastWriteTime -lt (Get-Date).AddDays(-30)} | Remove-Item

```

**## 🐛 HIBAELHÁRÍTÁS**

**### Gyakori hibák**

1. **\*\*Email küldés hiba\*\***: Outlook újraindítás

2. **\*\*DOCX olvasási hiba\*\***: Fájl integritás ellenőrzés

3. **\*\*Adatbázis lock\*\***: Kapcsolatok bezárása

**### Debug mód**

```robot

Log To Console    ${variable\_value}

Log    Detailed information    DEBUG

```

---

*\*🔧 Technikai dokumentáció - v2.1.0\**

*\*📅 Frissítve: 2025.08.25\**

## Melléklet:5 Telepítési útmutató:

**# 🤖 ROBOT FRAMEWORK PLÁGIUM ELLENŐRZŐ - TECHNIKAI ÁTTEKINTÉS**

**## 🎯 RENDSZER CÉLJA**

Automatizált DOCX dokumentum plágium ellenőrzés hash-alapú algoritmussal, Excel jelentéskészítéssel és automatikus email értesítéssel.

**## 🏗️ ARCHITEKTÚRA**

**### 🤖 Robot Framework Modulok**

- **\*\*PLG-00-main.robot\*\***: Főfolyamat vezérlő

- **\*\*PLG-02-read\_docx.robot\*\***: DOCX beolvasás és hash generálás

- **\*\*PLG-03-write-excel.robot\*\***: Excel export funkció

**### 🐍 Python Backend**

- **\*\*DocxReader.py\*\***: DOCX tartalom kinyerés

- **\*\*send\_email.py\*\***: Outlook COM email automatizálás

- **\*\*excel\_export\_simple.py\*\***: openpyxl alapú Excel generálás

- **\*\*plagium\_config.py\*\***: Konfigurációs osztály

**### 🗄️ Adatbázis Réteg**

- **\*\*SQLite\*\***: Helyi adatbázis (test\_database.db)

- **\*\*3 tábla\*\***: redundancia, hashCodes, repeat

- **\*\*Relációs kapcsolatok\*\***: Foreign key integritás

**## 🔍 PLÁGIUM ALGORITMUS**

**### 1️⃣ Hash Generálás**

```python

import hashlib

hash\_value = hashlib.sha256(line\_content.encode('utf-8')).hexdigest()

```

**### 2️⃣ Összehasonlítás**

```sql

SELECT file\_name, line\_content FROM hashCodes

WHERE hash\_value = ? AND file\_name != ?

```

**### 3️⃣ Kategorizálás**

- **\*\*🟢 Rendben\*\***: < 300 karakter redundancia

- **\*\*🟡 Gyanús\*\***: 300-1200 karakter redundancia

- **\*\*🔴 Másolt\*\***: > 1200 karakter redundancia

**## 📧 EMAIL AUTOMATIZÁLÁS**

**### Outlook COM Integráció**

```python

outlook = win32com.client.Dispatch("Outlook.Application")

mail = outlook.CreateItem(0)

mail.Send()  # Automatikus küldés

```

**### Hibakezelés**

- ✅ 3 próbálkozás exponenciális várakozással

- ✅ Piszkozat mentés fallback megoldás

- ✅ Részletes hibanaplózás

**## 📊 KIMENETEK**

**### Excel Jelentés**

- **\*\*Formátum\*\***: .xlsx (openpyxl)

- **\*\*Tartalom\*\***: Státusz, fájlnév, redundancia metrikák

- **\*\*Stílus\*\***: Színkódolt státusz, formázott táblázat

**### Email Template**

- **\*\*HTML formátum\*\***: Modern, reszponzív design

- **\*\*Melléklet\*\***: Excel fájl automatikus csatolás

- **\*\*Tartalom\*\***: Összesítő statisztikák + részletes lista

**## ⚙️ KONFIGURÁLÁS**

**### Plagium.config**

```ini

email=lovasz.otto@clarity.hu

input\_folder=d:\tmp

output\_folder=d:\tmp

email\_subject=Plagium Ellenorzes - Eredmenyek

excel\_prefix=plagium\_eredmenyek

```

**### Robot Framework változók**

```robot

${SQLITE\_DB\_FILE}    test\_database.db

${PYTHON\_EXEC}       rf\_env/Scripts/python.exe

${DOCUMENT\_PATH}     ${CONFIG\_INPUT\_FOLDER}

```

**## 🚀 FUTTATÁS**

**### Egyszerű futtatás**

```powershell

rf\_env\Scripts\robot.exe PLG-00-main.robot

```

**### Batch futtatás**

```robot

FOR    ${docx\_file}    IN    @{docx\_files}

    Fájladatok Rögzítése Redundancia Táblába    ${docx\_file}

    DOCX Beolvasás Teszt

END

```

**## 🔧 TELJESÍTMÉNY**

**### Feldolgozási sebesség**

- **\*\*Kis fájl\*\*** (<1MB): ~10-15 másodperc

- **\*\*Közepes fájl\*\*** (1-5MB): ~30-60 másodperc

- **\*\*Nagy fájl\*\*** (>5MB): ~2-5 perc

**### Memória használat**

- **\*\*Alapfolyamat\*\***: ~50-100 MB

- **\*\*Nagy dokumentum\*\***: ~200-500 MB

- **\*\*Batch feldolgozás\*\***: Lineáris skálázódás

**## 🛡️ BIZTONSÁG**

**### Adatvédelem**

- ✅ Helyi adatkezelés (nincs cloud)

- ✅ SHA-256 kriptográfiai hash

- ✅ Windows biztonsági modell

**### Hibakezelés**

- ✅ Try-except blokkok minden kritikus ponton

- ✅ Részletes error logging

- ✅ Graceful degradation

**## 📋 FÜGGŐSÉGEK**

**### Python csomagok**

```txt

robotframework==6.1.1

robotframework-databaselibrary==1.2.4

python-docx==0.8.11

openpyxl==3.1.2

pywin32==306

```

**### Rendszerkövetelmények**

- **\*\*OS\*\***: Windows 10/11

- **\*\*Python\*\***: 3.8+

- **\*\*Outlook\*\***: Microsoft Outlook telepítve és konfigurálva

- **\*\*Memória\*\***: Min. 4GB RAM

- **\*\*Tárhely\*\***: Min. 1GB szabad hely

**## 🔄 KARBANTARTÁS**

**### Adatbázis tisztítás**

```sql

DELETE FROM redundancia WHERE record\_date < date('now', '-90 days');

VACUUM;

```

**### Log fájlok rotáció**

```powershell

Get-ChildItem results\ -Name "\*.html" | Where-Object {$\_.LastWriteTime -lt (Get-Date).AddDays(-30)} | Remove-Item

```

**## 🐛 HIBAELHÁRÍTÁS**

**### Gyakori hibák**

1. **\*\*Email küldés hiba\*\***: Outlook újraindítás

2. **\*\*DOCX olvasási hiba\*\***: Fájl integritás ellenőrzés

3. **\*\*Adatbázis lock\*\***: Kapcsolatok bezárása

**### Debug mód**

```robot

Log To Console    ${variable\_value}

Log    Detailed information    DEBUG

```

---

*\*🔧 Technikai dokumentáció - v2.1.0\**

*\*📅 Frissítve: 2025.08.25\**

## Melléklet 6: Telepito.bat

@echo off

REM =========================================

REM  PLAGIUM ELLENORZO RENDSZER - TELEPITO

REM =========================================

setlocal EnableDelayedExpansion

echo.

echo =========================================

echo   PLAGIUM ELLENORZO RENDSZER TELEPITO

echo   Automatikus telepites es beallitas

echo =========================================

echo.

REM Telepitesi konyvtar bekeres

echo Adja meg a telepitesi konyvtar eleresi utjat:

echo (pl: C:\PlagiumEllenorzo vagy D:\MyProjects\PlagiumSystem)

echo.

set /p TARGET\_DIR="Telepitesi konyvtar: "

if "%TARGET\_DIR%"=="" (

    echo HIBA: Nem adott meg telepitesi konyvtarat!

    pause

    exit /b 1

)

echo.

echo Telepitesi cel: %TARGET\_DIR%

echo.

REM Ellenorizzuk a Python megletet

python --version >nul 2>&1

if errorlevel 1 (

    echo HIBA: Python nincs telepitve vagy nem elerheto!

    echo Kerem telepitse a Python 3.8+ verzioit a python.org oldalrol.

    pause

    exit /b 1

)

echo Python verzio:

python --version

echo.

REM Konyvtar letrehozasa ha nem letezik

if not exist "%TARGET\_DIR%" (

    echo Konyvtar letrehozasa: %TARGET\_DIR%

    mkdir "%TARGET\_DIR%"

    if errorlevel 1 (

        echo HIBA: Nem sikerult letrehozni a konyvtarat!

        pause

        exit /b 1

    )

) else (

    echo Konyvtar mar letezik: %TARGET\_DIR%

)

echo.

echo Fajlok masolasa...

REM Szukseges fajlok masolasa

copy "PLG-00-main.robot" "%TARGET\_DIR%\"

copy "PLG-02-read\_docx.robot" "%TARGET\_DIR%\"

copy "PLG-03-write-excel.robot" "%TARGET\_DIR%\"

copy "Plagium.config" "%TARGET\_DIR%\"

copy "plagium\_config.py" "%TARGET\_DIR%\"

copy "get\_config.py" "%TARGET\_DIR%\"

copy "send\_email.py" "%TARGET\_DIR%\"

copy "test\_database.db" "%TARGET\_DIR%\"

copy "excel\_export\_simple.py" "%TARGET\_DIR%\"

copy "TELEPITO\_UTMUTATO.txt" "%TARGET\_DIR%\"

REM Libraries mappa masolasa

if exist "libraries" (

    echo Konyvtarak konyvtar masolasa...

    xcopy "libraries" "%TARGET\_DIR%\libraries" /E /I /Y

)

REM Resources mappa masolasa

if exist "resources" (

    echo Eroforras konyvtar masolasa...

    xcopy "resources" "%TARGET\_DIR%\resources" /E /I /Y

)

REM Test mappa masolasa

if exist "test" (

    echo Teszt konyvtar masolasa...

    xcopy "test" "%TARGET\_DIR%\test" /E /I /Y

)

echo Fajlok sikeresen masolva.

REM Ellenorizzuk es javitsuk a hianyzo fajlokat

echo Hianyzo fajlok ellenorzese...

if not exist "%TARGET\_DIR%\plagium\_config.py" (

    echo plagium\_config.py hianyzo, ujra letrehozas...

    copy "plagium\_config.py" "%TARGET\_DIR%\"

)

echo.

REM Atlepunk a cel konyvtarba

cd /d "%TARGET\_DIR%"

REM Virtualis kornyezet letrehozasa

echo Virtualis kornyezet letrehozasa...

if not exist "rf\_env" (

    python -m venv rf\_env

    if errorlevel 1 (

        echo HIBA: Virtualis kornyezet letrehozasa sikertelen!

        pause

        exit /b 1

    )

    echo Virtualis kornyezet sikeresen letrehozva.

) else (

    echo Virtualis kornyezet mar letezik.

)

echo.

REM Virtualis kornyezet aktivalasa es csomagok telepitese

echo Csomagok telepitese...

rf\_env\Scripts\pip.exe install --upgrade pip

rf\_env\Scripts\pip.exe install robotframework

rf\_env\Scripts\pip.exe install robotframework-databaselibrary

rf\_env\Scripts\pip.exe install openpyxl

rf\_env\Scripts\pip.exe install python-docx

rf\_env\Scripts\pip.exe install pywin32

if errorlevel 1 (

    echo HIBA: Csomagok telepitese sikertelen!

    pause

    exit /b 1

)

echo.

echo start.bat fajl letrehozasa...

REM start.bat fajl letrehozasa

echo @echo off > start.bat

echo REM ========================================= >> start.bat

echo REM  PLAGIUM ELLENORZO RENDSZER FUTTATAS >> start.bat

echo REM ========================================= >> start.bat

echo echo. >> start.bat

echo echo ========================================= >> start.bat

echo echo   PLAGIUM ELLENORZO RENDSZER >> start.bat

echo echo   Main robot futtatas >> start.bat

echo echo ========================================= >> start.bat

echo echo. >> start.bat

echo. >> start.bat

echo REM Ellenorizzuk a virtualis kornyezet megletet >> start.bat

echo if not exist "rf\_env\Scripts\robot.exe" ^( >> start.bat

echo     echo HIBA: Virtualis kornyezet nem talalhato! >> start.bat

echo     echo Futtassa eloszor a telepito.bat fajlt! >> start.bat

echo     pause >> start.bat

echo     exit /b 1 >> start.bat

echo ^) >> start.bat

echo. >> start.bat

echo echo Konfiguracio ellenorzese... >> start.bat

echo if not exist "Plagium.config" ^( >> start.bat

echo     echo HIBA: Plagium.config fajl nem talalhato! >> start.bat

echo     echo Ellenorizze a konfiguracios fajlt! >> start.bat

echo     pause >> start.bat

echo     exit /b 1 >> start.bat

echo ^) >> start.bat

echo. >> start.bat

echo echo Robot Framework teszt futtatasa... >> start.bat

echo rf\_env\Scripts\robot.exe --outputdir results PLG-00-main.robot >> start.bat

echo. >> start.bat

echo if errorlevel 1 ^( >> start.bat

echo     echo HIBA: A teszt futtatasa sikertelen! >> start.bat

echo     echo Ellenorizze a log.html fajlt a reszletekert. >> start.bat

echo ^) else ^( >> start.bat

echo     echo. >> start.bat

echo     echo ========================================= >> start.bat

echo     echo TESZT SIKERESEN BEFEJEZODOTT! >> start.bat

echo     echo. >> start.bat

echo     echo Eredmenyek: >> start.bat

echo     echo - Log: results\log.html >> start.bat

echo     echo - Report: results\report.html >> start.bat

echo     echo - Email elkuldve a konfiguralt cimre >> start.bat

echo     echo ========================================= >> start.bat

echo ^) >> start.bat

echo. >> start.bat

echo pause >> start.bat

echo.

echo =========================================

echo TELEPITES SIKERES!

echo.

echo Telepitesi hely: %TARGET\_DIR%

echo.

echo Telepitett komponensek:

echo - Robot Framework

echo - Database Library

echo - OpenPyXL (Excel export)

echo - Python-docx (DOCX olvaso)

echo - PyWin32 (email kuldes)

echo - Teljes projekt fajlok

echo - start.bat futtato script

echo.

echo Hasznalat:

echo 1. Menjen a telepitesi konyvtarba: %TARGET\_DIR%

echo 2. Futtassa: start.bat

echo.

echo Konfiguracio: Plagium.config fajl szerkesztese

echo Tesztfajlok: test\ konyvtar

echo Eredmenyek: results\ konyvtar

echo =========================================

echo.

echo Szeretne most tesztelni a telepitett rendszert? (i/n)

set /p TEST\_NOW="Teszt futtatasa most: "

if /i "%TEST\_NOW%"=="i" (

    echo.

    echo Teszt futtatasa...

    cd /d "%TARGET\_DIR%"

    call start.bat

) else (

    echo.

    echo A rendszer keszen all a hasznalatra!

    echo Menjen a %TARGET\_DIR% konyvtarba es futtassa a start.bat fajlt.

)

echo.

pause