

Database Manager

Prog 3

Beadandó feladat

1. Osztályok leírása:

1.GUI_elements:(src mappa tartalma)

A. AddFrame:

- A tárolt adatok felvételét és szerkesztését segíti valósítja meg grafikusán a felhasználó számára.
- JDialog típusú elem, azért, hogy amíg az adatfelvétel/módosítás zajlik ne lehessen a main ablakban módosításokat végezni.
- Fájlok csatolását teszi lehetővé melyet a DocumentTable táblából kétszeres kattintással megnyithatunk.
- Csak .pdf típusú fájlok csatolhatóak.

- **Függvényei/Osztályai:**

- **public AddFrame(DatabaseFrame main, DocumentTable table, boolean toEdit):**

- A JDialog típusú AddFrame osztály konstruktora
 - létrehozza a dialog ablakot és a hozzá tartozó elemeket a meghatározott forma szerint.
 - **main:** DatabaseFrame típusú főablak referenciája
 - **table:** DocumentTable típusú változó az Adatokat tároló Táblázat
 - **toEdit:** boolean típusú változó, amely meghatározza, hogy szerkesztésre vagy új elem létrehozására lesz az ablak inicializálva

- **private void setBackAndForeGround(JComponent comp, Color backGround, Color foreGround):**

- Beállítja a Háttér és betűszínt a komponensnek
 - **comp:** JComponent típus
 - **backGround:** Color típus a háttérnek választott szín
 - **foreGround:** Color típus a betűszínnek választott szín

- **protected void loadValue():**

- Ha szerkesztésre lett létrehozva az ablak, akkor betölti a kiválasztott Dokumentum adatait
 - a megfelelő mezőkbe
 - Ha null értékű a kiválasztott elem (vagyis nem lett kiválasztva elem), akkor befejezi a művelet végzést

- **private boolean addDocumentToTable():**
 - Validáltatja a megadott adatokat és ha Valid akkor összeállítja a Felvételére küldendő Dokumentumot
 - **return: boolean** típusú Igaz, ha a visszatérési értéke, ha sikeres volt a Dokumentum felvétele, egyébként Hamis
- **private boolean updateDocumentAtTable():**
 - Validáltatja a megadott adatokat és ha Valid akkor összeállítja a Szerkesztésre küldendő Dokumentumot
 - **return: boolean** típusú Igaz, ha a visszatérési értéke, ha sikeres volt a Dokumentum szerkesztése, egyébként Hamis
- **private String[] createString():**
 - Visszaadja a Dialog mezőinek adataiból összeállított String-tömböt.
 - **return: String[]** típusú tömb az adatokból.
- **private void validateData() throws Exception:**
 - Validálja az ablak mezőinek értékét.
 - **Exception** típusú kivételt dob, ha valamelyik feltétel nem volt megfelelő a validáláshoz.
- **private class NameFieldVerifier extends InputVerifier:**
 - **InputVerifier** extension, osztály.
 - A megnevezés/azonosító mező validálására egy extension.
 - Levizsgálja, hogy a megadott id/név szerepel e már a modellben/táblában és ha igen akkor gátolja a fájl/dokumentum elmentést
- **private class AddFrameButtons implements ActionListener:**
 - AddFrameButtons belső osztály amely implementálja az ActionListener Interface-t
 - **ActionListener** interfészt implementál.
 - A Mentés és a Mégse gombok megnyomása esetén hívódik az actionPerformed override-olt függvénye
 - Belső paraméterektől függően vagy Elment/Szerkeszt vagy Bezárja az Ablakot

B. DatabaseFrame:

- A main Frame/Ablak, amelyen a felhasználó az egyes interakciógombokat, tárolt, megjelenített adatokat és menüt látja. Ezek megjelenítése és a kommunikáció a felhasználóval a feladata, amelyet JDialog segítségével valósít meg más elemek jelzésének hatására.
- Főbb részei/elemei: SidePanel, DocumentTable, dbMenu, FilterPanel
- **Függvényei/Osztályai:**
 - **public DatabaseFrame(String name):**
 - Az osztály default konstruktora, inicializálja az egyes elemeket és összeállítja az ablakot használatra.

- **name: String** az ablak megjelenő neve
- **protected boolean isFilterVisible():**
 - Visszaadja, hogy látható-e a szűrőpanel.
 - **return:** Igaz, ha a szűrőpanel látható, egyébként hamis
- **protected void showFilterer(boolean opt):**
 - Beállítja a szűrőpanel láthatóságát
 - **opt: boolean** típusú változó
 - Ha az értéke igaz akkor láthatóvá teszi a panelt.
 - Ha Hamis akkor elrejti.
- **protected int showDialog(String message, String title, String messageType):**
 - Megjeleníti a kapott üzenetet a megfelelő Dialógus formájában, hibaközlésre, illetve felhasználói értesítésre használatos.
 - **message:** a megjelenítendő üzenet
 - **title:** a Dialog ablak neve
 - **messageType:** az üzenet típusa
 - **return** ha párbeszéd ablak volt akkor a választott opció értéke, egyébként -1

C. DateLabelFormatter:

- Az AddFrame-en lévő dátumválasztó értékének megjelenítése, formázása a feladata, hogy a megfelelő formátumban jelenjen meg a kiválasztott dátum.
- **Függvényei/Osztályai:**
 - **public Date formatDate(String from) throws ParseException:**
 - Date típusúra formázza a kapott String-et
 - **throws** ParseException ha a formázás sikertelen
 - **from:** a formázandó dátum String formátumban
 - **return** Date típusú formázott érték
 - **public boolean isValidDateInterval(String from, String to) throws ParseException:**
 - Checkolja, hogy a kapott két dátum valid intervallumot határoz-e meg.
 - **from:** a kezdés dátuma String-ként meghatározva.
 - **to:** a befejezés dátuma String-ként meghatározva.
 - **return** Igaz, ha valid az intervallum, egyébként hamis
 - **throws** ParseException ha a formázás nem sikerült.

D. dbMenu:

- Szabványos menüpanel, amelyen főbb opciók a súgó/dokumentáció megnyitása, már mentett adatok betöltése (.ser formátum), vagy éppen a tárolt/szerkesztett adatok kimentése, összes adat törlése.
- A felhasználó számára könnyíti meg az egyes műveletek elvégzését

- Implementálja az ActionListener interfészt.
- **Függvényei/Osztályai:**
 - **public dbMenu(DatabaseFrame main, DocumentTable dataTable)**
 - Default konstruktor, ami létrehozza a menüpanelt
 - **main** a main ablak
 - **dataTable** az adatokat tároló tábla
 - a paraméterek az egyes műveletek és dialogok megjelenítéséhez elvégzéséhez szükségesek.
 - **private void setMenuItemDefaults(JMenuItem item, String actionCommand, ActionListener listener, ImageIcon icon):**
 - Beállítja az alap dolgokat az egyes menuItem-eknek
 - **item** a megjelenítendő icon ha van
 - **actionCommand** az actionCommand amit hozzárendelünk a menüponthoz
 - **listener** az ActionListener osztályunk ami a kiválasztásukat kezeli le.
 - **public void actionPerformed(ActionEvent e):**
 - Az egyes menüpontok kiválasztásának esetét kezeli le, és válassza ki a megfelelő opciót az actionCommand alapján.
 - **saveFile** kimenti a megadott névvel a fájlt (magát a típusát is utána kell írni a fájlnévnek(-.ser))
 - **loadFile** megpróbálja betölteni a kiválasztott .ser adatot
 - **deleteAll** kitörli a táblában lévő összes adatot
 - **helpDoc** megnyitja a dokumentációt

E. DefaultPaths:

- Enum osztály, amely az egyes Ikonok/Dokumentumok Working directory-n belüli helyét tartalmazzák.
- Feladata, hogy az egyes elérési útvonalakat összeállítsa az éppen aktuális working directory-től függően. És az így létrehozott path-eket kezelje/átadja átforgalmazva vagy szerkesztetlenül.
- **Függvényei/Osztályai:**
 - **DefaultPaths(String value):**
 - Konstruktor, ami beállítja az adott enum pathját.
 - **value** az elérési útvonal.
 - **getter setter metódusok az egyes típusok visszaadásához.**

F. DocumentTable:

- A tárolt Dokumentumok/Adatok megjelenítését, megnyitását szerkesztését törlését, szűrését valósítja meg, JTable típusú osztály mely választás szempontjából kézenfekvő volt.
- **Függvényei/Osztályai:**

- **public DocumentTable(DatabaseFrame mainframe):**
 - Default konstruktor, ami a táblázat a hozzátartozó modell és sorter létrehozását valósítja meg.
 - Formázza a táblázatot és beállítja a megfelelő szempontokat
 - **mainframe** a main Ablak referenciája
- **public boolean isCellEditable(int row, int column):**
 - Letiltja a cellák szerkesztését
- **protected boolean checkIDValidity(String docID):**
 - az Hashtábla key mezőjének megdandó érték validitását vizsgálja.
 - **docID** megadni kívánt id
 - **return:** Igaz ha valid a megadni kívánt id, vagyis elfogadható
- **protected void addRow(String[] doc):**
 - Új Sor/Dokumentum hozzáfűzése a modellhez.
 - Létrehozza a dokumnetumot majd hozzáadja a modellhez.
 - **doc** az új felvenni kívánt dokumentum szöveges állomány formájában
- **protected void editRow(Document doc, String oldID):**
 - A kijelölt sor szerkesztését hajtja végre.
 - Ha nem sikerült valamilyen oknál fogva azt jelzi a mainFrame-nek,
 - aki dialoggal közli a felhasználóval a bekövetkezett hibát
 - **doc** az új dokumentum
 - **oldID** a régi dokumentum azonosítója
- **protected void removeSelectedRow():**
 - kitörli a kijelölt sort/elemet a modelljéből
- **protected void saveData(String savedFile):**
 - A Modell tartalmát menti ki a felhasználó által meghatározott fájlba
 - aminek a kiterjesztése .ser kell legyen.
 - Ha szűrési feltétel van alkalmazva akkor az az alapján megjelenő elemek kerülnek kimentésre.
 - **throws** Exception ha üres modellt/szűrőt akarunk kimenteni
 - Sikertelen mentés esetén azt dialoggal jelzi a mainFrame-nek aki pedig a felhasználónak.
 - **savedFile** a kimenteni kívánt fájl elérési útvonala

- **private ArrayList<String> getVisibleData():**
 - Visszaadja az éppen aktuálisan modellben lévő elemek idjait a mentéshez.
 - Ennek a segítségével lehet kimenteni a szűrési eredményként kapott Dokumentumokat
 - **return** a megjelenített id-k
- **protected void loadData(String loadFile):**
 - Betölti a DocumentTable-be a .ser típusú szerializált fájlt
 - Csak akkor tölti be fájlt ha az a felhasználó is megerősíti és a tárolt adatok elvesznek ekkor.
 - Ha sikertelen volt a betöltés azt a felhasználónak jelzi.
 - **loadFile** a betölteni kívánt fájl elérési útvonala
- **protected void clearAll() :**
 - Kitörölteti az összes adatot a modellből
- **protected void addRowFilter(RowFilter<DocumentTableModel,Integer> newRowFilter):**
 - Hozzáadja a sorfiltert a táblához szűrés esetén hívódik.
 - Az átadott szűrő tartalmazza a legújabb szűrési feltételt.
 - **newRowFilter:** az újonnan meghatározott szűrés.
- **protected void openFile(int rowID):**
 - Megyinitja a csatolt fájlt ha egyáltalán van olyan
 - Ha nincs akkor pedig hibaüzenetet dob
 - Nem talált fájl esetén is NullPointerException-t dob
 - **rowID** a sorazonosító, amely a kijelölt sort jelenti
- **protected Document getSelectedRowData():**
 - Visszaadja a Modellből a kiválasztott sor adatait.
 - **return** a kijelölt Document típusú változó

- **private class CellDbClick extends MouseAdapter:**

- kattintás esetén hívódik meg.
- Privát osztály mely a MouseAdapter osztályból származik.
- Kétszeres kattintás esetén megnyitja a csatolt fájlt, ha van olyan
- egyébként üzenettel jelzi, ha nincs csatolmány
- Egyéb esetben pedig, ha sor lett kijelölve akkor meghívja az optpanel (SidePanel) setDeletable metódusát

G.FilterPanel:

- A DocumentTable típusú táblában tárolt adatok szűrését segíti/valósítja meg grafikusan.
- Szűrési szempont lehet:
 - név
 - típus
 - Kezdet:
 - Elfogadott formátum: yyyy.MM.dd vagy yyyy/MM/dd
 - Intervallum megadása az elfogadott formátumokban „-” jellel elválasztva valósítható meg. Ahol az első dátum az intervallum kezdetét a második dátum az intervallum végét jelöli. Pl.:
2020.10.01-2020.11.01
- Vége: Hasonlóan a Kezdet mezőhöz.
- **Függvényei/Osztályai:**
 - **public FilterPanel(DatabaseFrame main,DocumentTable table):**
 - Default konstruktor.
 - Létrehozza és inicializálja a szükséges elemket, a filtert, és formázza a panelt
 - **main:** a Fő ablak
 - **table:** az adatokat tároló DocumentTable
 - **private void setBackAndForeground(JComponent comp):**
 - Beállítja a színeket a kapott komponensnek, alapértelmezett belső változó a beállítandó szín.
 - **comp** a kapott komponens

- **private void setLayoutToPanel(JPanel panel,int top,int left,int bottom, int right,int AXIS):**
 - Beállítja a layout-ot és a formázást a paneleknek
 - **panel** a formázandó panel
 - **top** igazítás fent
 - **left** igazítás balra
 - **bottom** igazítás lent
 - **right** igazítás jobbra
 - **AXIS** a layout iránya/eloszlása
- **private void defineLabel(JLabel label,String ToolTipText):**
 - Inicializálja a paraméterként kapott labelt
 - **label** a formázandó label
 - **ToolTipText** "good to know" szöveg / ha van
- **private void defineTextField(JTextField textField, ActionListener listener):**
 - beállítja a paraméterként kapott textfield-et.
 - **textField** formázandó field.
 - **listener** Action listener amit az adott fieldhez kell adni.
- **private class FilterAction implements ActionListener:**
 - Belső osztály, ami implementálja az ActionListener interface-t
 - A mező változásakor, ami enter leütést jelöl, hívódik meg és illeszti össze a filterer segítségével a szűrendő adatokat majd átadja a táblának szűrésre.

H.SidePanel:

- Az akciógombokat és a hozzájuk tartozó funkcionalitás elindítását megvalósító függvényeket/listenereket tárolja/valósítja meg.
- JPanel típusú osztály
- Funkciógombok:
 - hozzáadás: új elem felvétele AddFrame megnyitása/létrehozása
 - törlés: a Table-ben kijelölt elem törlése, Disabled, ha nincs kijelölt elem.
 - keresés: A kereső panel FilterPanel megjelenítése/elrejtése
 - szerkesztés: a kiválasztott elem szerkesztése az AddFrame ablakban. Disabled, ha nincs kiválasztott elem.
- **Függvényei/Osztályai:**
 - **public SidePanel(DatabaseFrame main, DocumentTable tab):**
 - SidePanel default konstruktora amely létrehozza a panelt a rajta található nyomógombokkal és azok funkcióival.

- Beállítja a default GUI elemeket is.
- **main** a Fő Ablak mainFrame
- **tab** az Adatokat tároló Tábla (DocumentTable)
- **private void setButtonDefault(JButton button, Color backGround, Color foreGround, ActionListener listener, String ac, boolean enabled):**
 - Beállítja a nyomógombokhoz tartozó default paramétereket, és tulajdonságokat.
 - **button:** a nyomógomb
 - **backGround:** a hozzá tartozó háttérszín
 - **foreGround:** a hozzá tartozó betűszín
 - **listener:** az ActionListener amely a nyomógomb megnyomását figyeli
 - **ac:** actionCommand paraméter amely alapján beazonosítható mely gomb lett megnyomva.
 - **enabled:** ha Igaz akkor alapértelmezett aktivált a nyomógomb, egyébként disabled.
- **protected void setDeletable(boolean opt):**
 - Használhatóvá teszi vagy letilja a szerkesztés és törlés gombokat
 - **opt** a beállítandó opció: igaz vagy hamis
- **public class PanelButton implements ActionListener:**
 - Az akció gombok megnyomása esetén hívódik meg.
 - **add** esetén létrehozza az AddFrame-et hozzáadásra
 - **delete** esetén meghívja a DocumentTable removeSelectedRow metódusát
 - **edit** esetén létrehozza az AddFrame-et szerkesztésre
 - **search** esetén meghívja a DatabaseFrame isFilterVisible és a showFilterer metódust

2.Logical_elements:(src mappa tartalma)

A. DateMatcher:

- Logikai osztály, amely az egyes dátum formátumok/ dátum intervallum formátumok validálását, formázását valósítja meg.
- A FilterPanel-ben megadott dátum paramétereket validálja, és jelzi, ha nem megfelelő a formátum.
- **Függvényei/Osztályai:**
 - **public boolean matches(String date,String type):**
 - megvizsgálja, hogy a megadott string megegyezik e meghatározott formának
 - **date** a vizsgálandó szöveg /String
 - **type** a meghatározott formátum
 - **return** Igaz, ha az adott formátumra egyezik a kapott szöveg patternje
 - **public boolean matchInterval(String interval, String type):**

- megvizsgálja, hogy a megadott string megegyezik-e a meghatározott formának
 - **date** a vizsgálandó szöveg /String
 - **type** a meghatározott formátum
 - **return** Igaz, ha az adott formátumra egyezik a kapott szöveg patternje
- **public LocalDate format(String from) throws ParseException:**
 - formázza a kapott stringet az előzetesen validálásnál beállított dateType alapján.
 - **from** a formázandó szöveg
 - **return** A formázott Dátum LocalDate formátumban
 - **throws** ParseException ha a formázás nem sikerülne
 - **public boolean validateDate(String dateText):**
 - Megvizsgálja a kapott stringet, hogy illeszkedik-e vagy a dátum patternekre, vagy pedig a dátum intervallum patternekre. Ha egyezik akkor beállítja a dateType változóját az egyező minta azonosítójára.
 - **dateText** a vizsgálandó szöveg
 - **return** Igaz, ha van egyezés valamelyik pattern-re

B. DBmain:

- A main osztály
- létrehozza a DatabaseFrame-et és megjeleníti azt

C. Document:

- Az egész Project által kezelt Dokumentumtípus osztálya
- Ezt jeleníti meg a DocumentTable
- Ezeket szűri a FilterPanel
- Ezeket lehet szerkeszteni és törölni, és ha van hozzá csatolt fájl akkor azok megjelenését/megnyitását elvégezni.
- Főbb azonosítói:
 - Név/ID: A meghatározott Dokumentum neve
 - Típus: A típusa
 - Kezdés: Érvényesség kezdete
 - Vége: Érvényesség vége
 - Fájl: a Csatolt fájl neve, ha van csatolva, egyébként a „Nincs csatolmány” szöveg jelenik meg.

• Függvényei/Osztályai:

- **public Document(String[] input):**
 - A Document osztály default konstruktora amely a Dokumentum objektum létrehozásáért felel
 - Elmenti az adatokat és csatolja a fájlt az objektumhoz, ha van fájl input az eltárolandó adat string tömbként

- **input:** az értékül adandó attribútumokat és fájl elérési útvonalat tartalmazza/ ha van.
- **protected Object[] toObjectArray():**
 - Megformálja a Dokumentum osztály paramétereit úgy, hogy az a DocumentTableModel által megjeleníthető legyen
 - **return** Objektum tömb ami a Dokumentum adatait tartalmazza
 - kivéve a fájl mert annál a neve, ha van csatolt fájl
 - "Nincs csatolmány", ha nincs
- **protected boolean containsFile():**
 - Visszaadja, hogy tartalmaz e csatolmányt az objektum
 - **return** Igaz, ha van csatolmány és létezik.
- **protected void openDoc(Desktop desktop) throws Exception:**
 - Megnyitja a csatolt fájlt, ha van.
 - desktop Desktop az asztalt reprezentáló objektum
 - **throws** Exception ha nem tudja megnyitni a dokumentumot <=> nincs csatolmány

D. DocumentFilterer:

- Logikai osztály, amely a FilterPanel-ben meghatározott értékek alapján végzett keresést valósítja meg a DocumentTable-be betöltött elmentett állományok/dokumentumok között.
- a mezők típusától: LocalDate, String függően állítja össze a filtert, amit aztán továbbít a táblának
- **Függvényei/Osztályai:**
 - **public DocumentFilterer():**
 - Default Konstruktor, ami inicializálja a dateMatcher-t és a filterList-et, melyek az osztály belső attribútumai, bővebben javaDoc
 - **public void clearFilterer():**
 - kiüríti a szűrőfeltétel listát
 - **public void add(String text,int column,String field) throws Exception:**
 - hozzáad egy új filtert a filterListhez a meghatározott feltételek alapján.
 - Validál és ha minden dolog megfelelő akkor létrehozza a szűrőt amit utána felvesz a filterListbe.
 - **text** szűrendő szöveg/dátum a típustól meghatározva függően
 - **column** a szűrendő oszlop azonosítója.
 - **field** a field típusa amit szűrünk
 - **text** a szűrendő field egyszerű szöveg
 - **date** a szűrendő field dátum

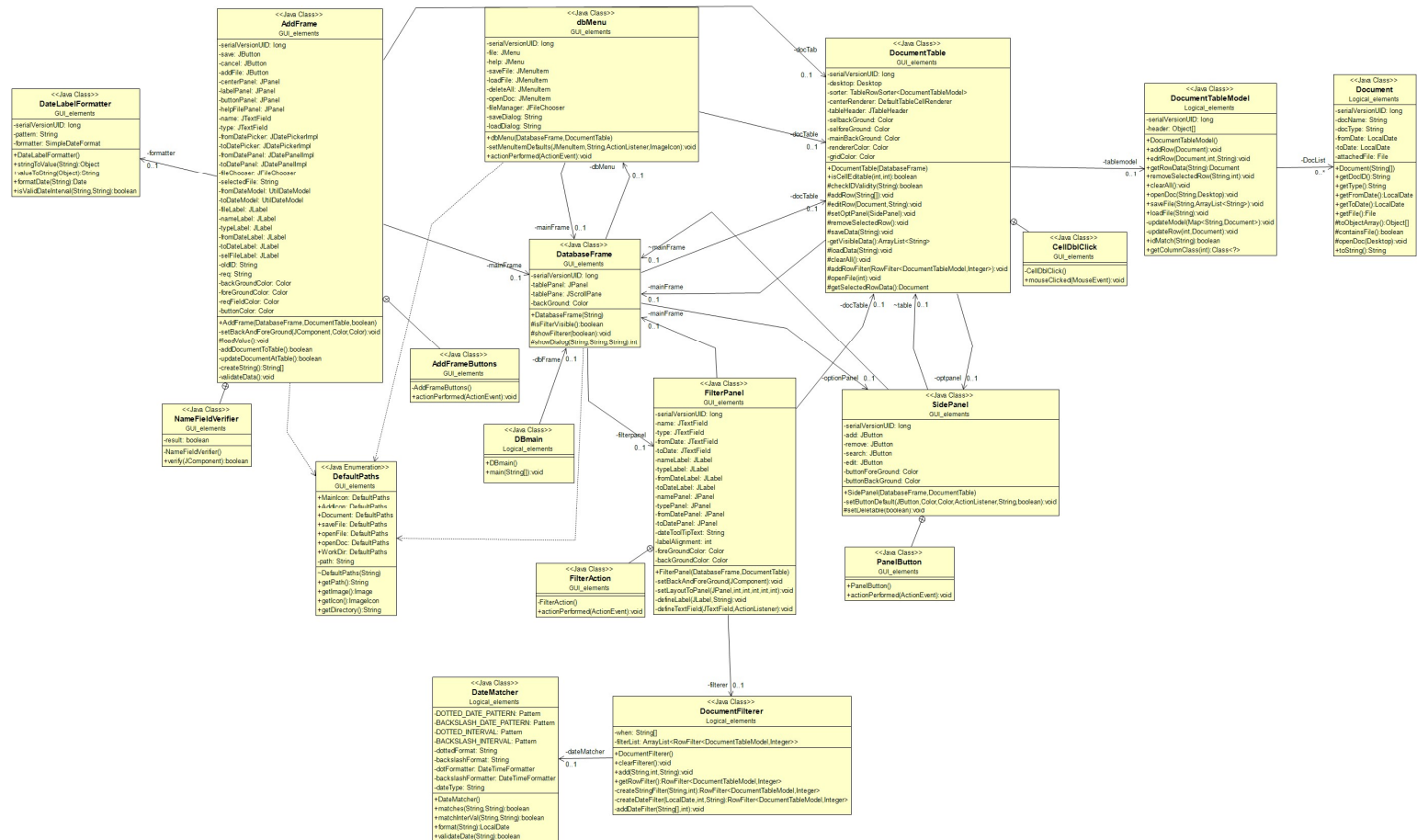
- **throws** Exception ha a mező nem beazonosítható vagy ha nem megfelelő a megadott dátum formátum
- **public RowFilter<DocumentTableModel,Integer> getRowFilter():**
 - Létrehoz egyetlen filtert and kapcsolattal a afilterListben meghatározott elemekből
 - **return** filter a filterList elemeiből
- **private RowFilter<DocumentTableModel,Integer> createStringFilter(String regex, int where):**
 - létrehoz egy szövegeket szűrő filtert String filtert
 - **regex** a szűrendő/keresendő szöveg/szövegrészlet
 - **where** az oszlop/column neve, ahova a filter kiértékelődik
 - **return** String filterer a megadott szöveg/szövegrészletből
- **private RowFilter<DocumentTableModel,Integer> createDateFilter(LocalDate date,int where,String type):**
 - Létrehoz egy LocalDate típusú dátumokat szűrő szűrőt/filterert a megadott paraméterek alapján
 - **date** a szűrendő dátum
 - **where** melyik oszlopra értékelendő ki a filter
 - **type** a szűrési feltétel
 - **before** azon dátumok szűrése amelyek korábbiak mint a date-ben meghatározott dátum
 - **after** azon dátumok szűrése amelyek későbbiek mint a dateB-ben meghatározott dátum
 - **return** LocalDate típusú elemeket szűrő filter
- **private void addDateFilter(String[] dates, int where) throws ParseException:**
 - A kapott dátumokat tartalmazó String tömböt formázza
 - létrehozza a dátum szűrőt a megfelelő oszlopra, majd hozzáadja a filterListhez
 - **dates** a szűrendő dátumok
 - **where** melyik oszlopra/column-ra
 - **throws** ParseException ha a fordítás nem volt sikeres.

E. DocumentTableModel:

- DocumentTable által a megjelenítésre és az adatokkal való műveletek elvégzésére használt modell.
- Ez tárolja ténylegesen a betöltött Dokumentumokat, végzi a kimentésüket/ betöltésüket törlésüket és szerkesztésüket a modellből/modellben és a táblában

- Azért döntöttem külön modell definiálása mellett, mivel a csatolt fájlokat nem lehet szépen megjeleníteni a táblázatban, és az adatok keresése, sem a leggyorsabb, ezért a betöltött Dokumentumokat egy HashTable-ben tároljuk a nevével/id-val azonosítva. Amelyet formázva töltünk be mentésnél szerkesztésnél a táblázat által használt modellbe.
- **Függvényei/Osztályai:**
 - **public DocumentTableModel():**
 - Default konstruktor ami létrehozza a modellt és inicializálja a Doclist-et, mely a Documentumokat tartalmazza
 - bővebben javaDoc
 - **public void addRow(Document doc):**
 - Dokumentum hozzáadása a listához és a modellhez
 - **doc:** a Hozzáadandó Dokumentum.
 - **public void editRow(Document newDoc, int viewRowID, String oldID) throws Exception:**
 - A Dokumentum szerkesztésénél hívódik meg, az új dokumentumot hozzáadja a listához és a modellhez a régit pedig törli
 - **newDoc** a felvevendő dokumentum
 - **viewRowID** a modell ViewModelljének a dokumentumhoz tartozó id-ja
 - **oldID** a régi dokumentum id-ja/neve
 - **throws Exception** ha nem találta meg az oldID alapján az adott elemet, vagyis nincs a listában.
 - **public Document getRowData(String docID):**
 - Visszaadja az adott Dokumentumot ami a modellben kiválasztva szerepel
 - **docID:** a kiválasztott dokumentum neve/id ja
 - **return** Document a kiválasztott dokumentum
 - **public void removeSelectedRow(String rowID,int row) throws NullPointerException:**
 - Kitörli a megfelelő sort a modellből és a listából is.
 - **rowID** a dokumentum azonosítója
 - **row** a modell sor azonosítója
 - **throws NullPointerException** ha nem található a modellben a dokumentum
 - **public void clearAll():**
 - Kiüríti a listát és a modellt

- **public void openDoc(String docID, Desktop desktop) throws Exception:**
 - A modellben kiválasztott elemre kétszer kattintva meghívódik a DocumentTable-ön keresztül
 - megnyitja a csatolt fájlt / ha van olyan
 - **docID** a dokumentum azonosítója
 - **desktop** az asztalt jelképező objektum
 - **throws** Exception ha nincs csatolt állomány
- **public void saveFile(String savedFile, ArrayList<String> ids) throws IOException:**
 - Az ids tömb paramétereivel megegyező mezők/dokumentumok tartalmát tudjuk kimenteni a megadott fájlba
 - **ids** a modellben éppen aktuálisan levő fájlok id-jai
 - **savedFile** a fájl absolute path-ja
 - **throws** IOException, ha valamilyen oknál fogva nem sikerült volna a fájlba írás.
- **public void loadFile(String loadFile) throws IOException, ClassNotFoundException:**
 - Megpróbálja betölteni szerializálni a megadott fájlt, amit betölt a listába és a modellbe is.
 - **loadFile** a betöltendő fájl path-je
 - **throws** IOException ha nem sikerült a betöltés.
 - **throws** ClassNotFoundException ha a kasztolás sikertelen volt.
- **private void updateModel(Map<String, Document> map):**
 - frissíti/felül írja a listát és a modellt is.
 - **map** a betöltött állomány
- **private void updateRow(int rowID, Document doc):**
 - A modell egy sorának megváltozott/szerkesztett sorának értékét frissíti
 - **rowID** a változtatott sor id-je
 - **doc** a megváltozott Dokumentum
- **public boolean idMatch(String wannabeID):**
 - visszaadja, hogy van e a wannabeID hoz hasonló
 - az adat szerkesztés és létrehozás vizsgálatához kell.
 - **wannabeID** a vizsgálandó id
 - **return** Igaz, ha van már hozzá hasonló id
- **public Class<?> getColumnClass(int column):**
 - osztály vizsgálat ami a filter kiértékelésnél szükséges.
 - **column:** a JTable éppen aktuális oszlópa.



2. JUnit tesztek leírása:

1. Logical_elements:

A. DocumentTableModelTest:

- A DocumentTableModel és a köztes függvények tesztelésére létrehozott tesztosztály.
- **Függvényei/Osztályai:**
 - **public void SetUp():**
 - Az inicializálás az alap paramétereknek, melyek a tesztelés elvégzéséhez szükségesek.
 - **public void testAddRemoveRow():**
 - addRow és removeSelectedRow vizsgálata, hibamentes lefutást kellene megvalósítania.
 - **public void testEditRow() throws Exception :**
 - EditRow kivételkezelésének vizsgálata
 - Exception-t dob hiszen nem létező indexnél akarunk elemet cserélni.
 - **public void testEditRow2() throws Exception:**
 - Sikeres szerkesztés vizsgálat
 - Ha létező indexel hívjuk meg az editRow függvényt és az értékváltozás vizsgálata
 - **public void testOpenDoc() throws Exception:**
 - Nemlétező Dokumentumindex-hez tartozó fájl megnyitása
 - Exception dobás a várt kimenet
 - **public void testOpenDoc2()throws Exception:**
 - Fájl megnyitás vizsgálata
 - Nincs csatolt fájl így Exception hibát kell dobnia
 - **public void SaveAndLoadTest() throws Exception:**
 - A fájl mentésének és betöltésének vizsgálata
 - Sikeres lefutást kell produkálnia.

B. DocumentFiltererTestValidData:

- Az Szűrésnél alkalmazott validáló és filter készítő tesztelése valid, megfelelő adatokkal, közvetve a DateMatcher-t is teszteljük.
- **Függvényei/Osztályai:**
 - **public class DocumentFiltererTestValidData:**
 - Teszteset a filterek valid paraméterrel történő létrehozására.
 - Az osztály definiálja és egyesével paraméterül adja a paramétereket a testfüggvényünknek.

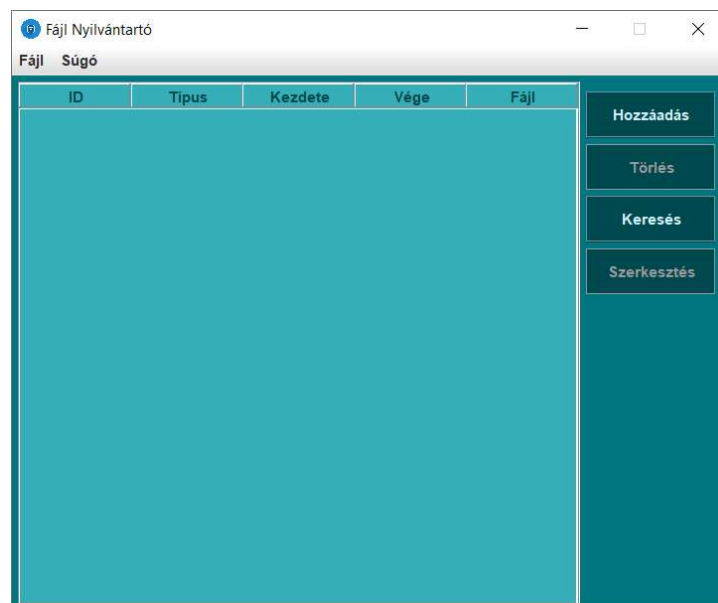
- ennél a tesztesetnél minden paraméterrel le kell tudni futnia Exception hiba nélkül a kódnak.
- **public DocumentFiltererTestValidData(String str)**
 - Az osztály default konstruktora ami egyesével paraméterül veszi a definiált paraméter tömb elemeit.
- **public void testAdd()throws Exception:**
 - Ez a függvény teszteli az egyes bemeneti paraméterek validak-e.

C. DocumentFiltererTestInvalidData:

- Az Szűrésnél alkalmazott validáló és filter készítő tesztelése rossz adatokkal, hasonlóan itt is Teszteljük a DateMatcher metódusait is közvetve.
- Célunk, hogy rossz eredményekre Exception-t dobjon a program.
- **Függvényei/Osztályai:**
 - **public void SetUp():**
 - Az inicializálás az alap paramétereknek, melyek a tesztelés elvégzéséhez szükségesek.
 - **public void TestInvalid1() throws Exception**
 - Rossz dátum típusok, azért teszteltem őket külön mert paraméteresen az első után EXCEPTION miatt kilép.
 - A nem megfelelő formátumok tesztelése és azok exception dobásának figyelése 5 db függvénnyel különböző bemeneti típusokra.
 -
 - **public void TestInvalid5() throws Exception**

3. Felhasználói kézikönyv:

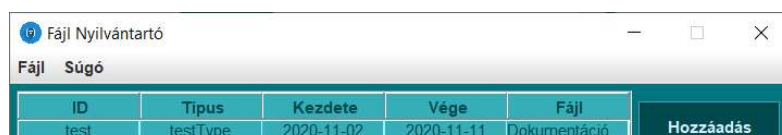
1. A fájl megnyitása után a következő ablak fogadja a felhasználót:



2. Itt a következő lehetőségek közül választhat:

A. Az oldalmezőn a Hozzáadás gombbal felvesz új elemet ekkor egy másik ablak nyílik meg ahol az egyes adatokat értékeket megadva, hozzáadhatja a fájlt a nyilvántartóhoz. Itt a következőkre kell figyelni:

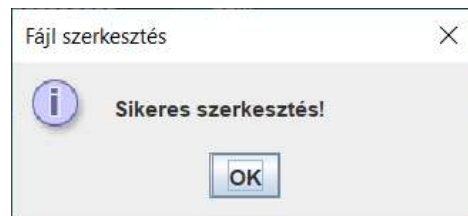
- nem adható meg már használatban lévő azonosító/vagy üres azonosító, nem hagyható üresen a dátum kezdete és vége mező.
- Fájl csatolása gomb megnyomása után egy fájlválasztó ablak jelenik meg, amelyben kiválaszthatja a felhasználó, ha kíván csatolni állományt, az új bejegyzéshez. Ha van csatolt állomány akkor annak neve a „Fájl:” részen jelenik meg. Az adat felvételt a mentés gomb megnyomásával lehet permanensé tenni, ekkor az új rekordot felvesszük az adatbázisba, és megjelenítjük a felhasználónak.
- Mégse gomb megnyomása esetén a megadott paraméterek elvesznek.



ID	Típus	Kezdete	Vége	Fájl	
test	testType	2020-11-02	2020-11-11	Dokumentáció...	Hozzáadás

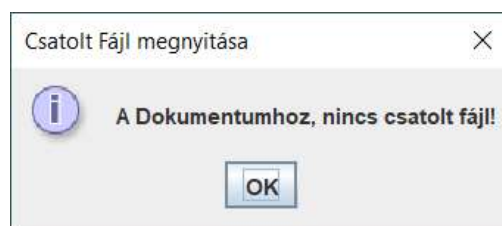
B. Az adatbázisban már felvett adatokat, lehet szerkeszteni, törölni, csatolt fájlok esetén ezeket megnyitni, illetve keresni közöttük különböző paraméterek alapján.

- Törlés és Szerkesztés egészen addig inaktív, amíg ki nem választjuk a törölni/szerkeszteni kívánt elemet a listából.
- Szerkesztés esetén a kiválasztott elem adatait betöltjük a Hozzáadásnál is látott ablakba, ahol szerkeszteni lehet az adat egyes részeit, a hozzáadással kapcsolatos megszorítások ebben az esetben is érvényesek.
- Sikeres szerkesztés esetén a következő ablak tájékoztatja arról a felhasználót:



C. Csatolt fájl megnyitása a megnyitni kívánt dokumentum kétszeres kattintásával, érhető el:

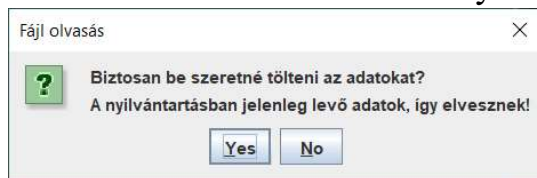
- ekkor a rendszerben alapértelmezett adott fájlpushoz tartozó szerkesztő ablak nyitja meg a csatolt fájlt, vagy ha nincs csatolt állomány akkor a következő ablak jelenik meg:
- Esetleges hiba bekövetkezését a program hasonló dialógus ablakokkal jelzi.



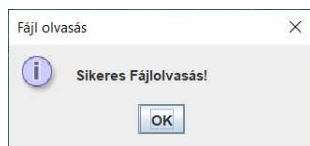
D. Lehetőségünk van már felvett adatok mentésére, betöltésére is, ezt a menüpanelen a Fájl menüpont alatt találhatjuk meg:



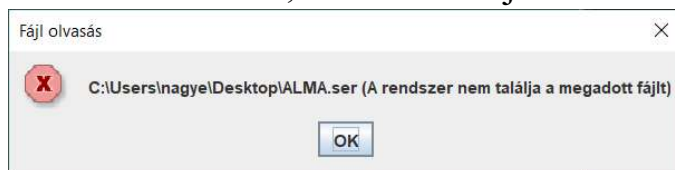
- Az „Adatok betöltése” / „Adatok mentése” menüpontok választása esetén egy fájlválasztó ablak jelenik meg ahol kiválaszthatjuk/megadhatjuk a betölteni kívánt állományt.
- Fontos megkötés az állományra vonatkozóan, hogy csak .ser típusú fájlok tölthetők be a rendszerbe.
- Betöltés esetén az éppen az adatbázisban tárolt elemek törlődnek, így itt a rendszer rákérdez a betöltés folytatására:



- Sikeres mentés, betöltés esetét az alkalmazás dialógussal jelzi, például:



- Sikertelen betöltés, nem létező fájl betöltésének esetét, is hasonlóan jelzi.



- Lehetőségünk van a nyilvántartásban tárolt összes adat törlésére is, a Fájl/Összes törlése menüpont választásával.
- A Súly menüpont alatt pedig ez a Dokumentáció nyitható meg, további tájékoztatás, segítség céljából.

E. A már felvett adatokat tudjuk elemenként rendezni, illetve szűrni is, valamely tulajdonság(aki) alapján:

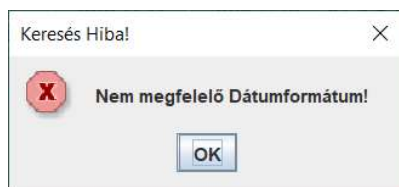
- Rendezni a táblázatban a rendezés tárgyául használni kívánt attribútum fejlécére kattintva lehet, növekvő, vagy csökkenő sorrendben, egyszerre csak egy attribútum alapján.





- A fájlok szűrésére szolgáló szűrőpanelt, a Keresés gomb megnyomásával jeleníthetjük meg a táblázat alatt, vagy tüntethetjük el.

- A Keresés bármely paraméterek megadásával, és Enterrel történő nyugtázásával végezhető el, üres mező esetén a keresés nem értékelődik ki az adott paraméterre.
- Megkötések:
 - Dátumot csak:
 - yyyy.MM.dd
 - yyyy/MM/dd
 - formában adhatunk meg/fogad el a filter.
 - Megadhatunk a dátumoknak keresési intervallumot is, hogy mely időponttól kezdődően, mely időpontig bezárólag szűrje azokat.
- További megkötés erre vonatkozólag, hogy az adott mezőbe csak a két dátum kerüljön a fent meghatározott formátum valamelyikében „-” jellel elválasztva szóközők nélkül, formailag tehát:
 - yyyy.MM.dd-yyyy.MM.dd
 - yyyy/MM/dd-yyyy/MM/dd
- Sikertelen szűrés esetén a program, dialógussal jelez, rossz/hibás formátum esetén is:



- Sikeres szűrés esetén a szűrési feltételek kiértékelődnek az adatokra, és azon adatok jelennek meg, amelyek a feltételeknek megfelelnek, ezeket szerkeszteni, kimenteni és törölni is lehet.

Fájl Nyilvántartó

Fájl Súly

ID	Típus	Kezdete	Vége
test	testtype	2020-11-01	2022-11-16
test1	alma	2020-11-11	2024-11-07

Azonosító:

test

Típusa:

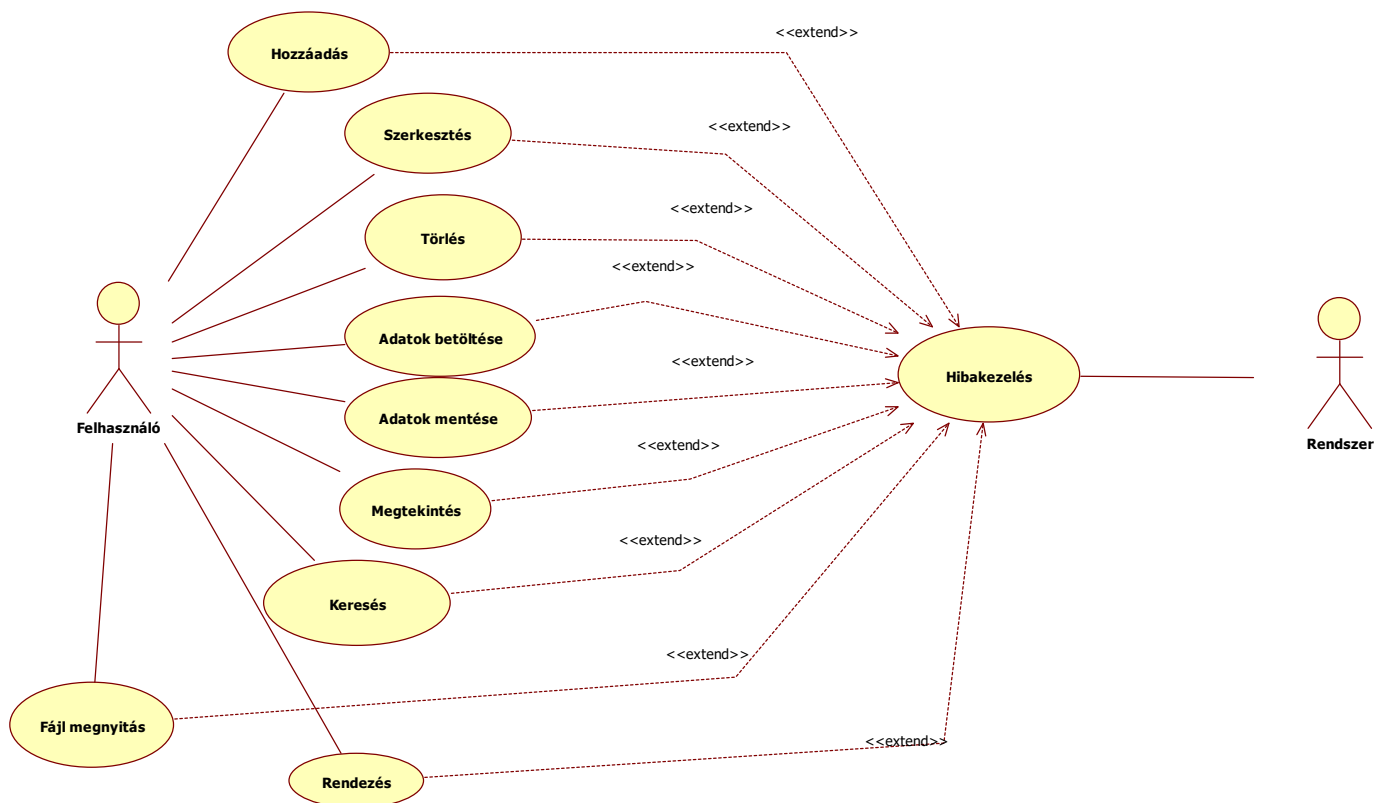
Kezdete:

2017.08.02

Vége:

4. Use-case -k:

1. Ábra:



2. Use Case -ek leírása:

1. Hozzáadás:

Cím	Adatok hozzáadása
Leírás	A felhasználó egyesével tud értéket adni az adatbázishoz
Aktorok	Felhasználó
Főforgatókönyv	1. A felhasználó megadja az attribútumokhoz tartozó adatokat, amelyeket elszeretne tárolni az adatbázisban, majd elmenti azt.
Alternatív forgatókönyv	1.A.1 A felhasználó megadja az attribútumokhoz tartozó adatokat, de hibásan, vagy a kötelező adatok közül nem megfelelő formátumban vagy hiányosan adta meg adatot, így a mentés nem megy végbe, és elindul a hibakezelés.

2. Szerkesztés:

Cím	Adat módosítás
Leírás	A felhasználó módosítani tudja a már elmentett rekordokhoz/dokumentumhoz tartozó értékeket.
Aktorok	Felhasználó
Főforgatókönyv	1. A felhasználó megadja a módosítandó értékeket, majd mentéssel konstatálja döntését.
Alternatív forgatókönyv	1.A.1 A felhasználó megadja az elmenteni kívánt értékeket, de minden kötelező adatot helyesen és/vagy üresen határozza meg azokat, így a mentés nem megy végbe és elindul a hibakezelés

3. Törlés:

Cím	Adat Törlés
Leírás	A felhasználó a nyilvántartásba vett értékek közül tud törölni rekordokat.
Aktorok	Felhasználó
Főforgatókönyv	1. A felhasználó kiválasztja a törlendő rekordot, majd a törlés gomb megnyomásával törli azt.
Alternatív forgatókönyv	1.A.1 A törlés során esetlegesen a felhasználó tevékenységétől függetlenül bekövetkező hiba, elindítja a hibakezelést

4. Adatok betöltése:

Cím	Mentett adatok betöltése
Leírás	A felhasználó az előzetesen kimentett adatokat tudja betölteni a nyilvántartó rendszerbe, további használat céljából.
Aktorok	Felhasználó
Főforgatókönyv	1. A felhasználó a menüopciók közül kiválasztja az „Adatok betöltése” nevezetű opciót, majd meghatározza a betölteni kívánt

	<p>fájlt, a felugró dialog ablakban elfogadja, hogy a már tárolt adatok más adatok betöltésével törlődnek és betölti azt.</p> <p>2. A felhasználó a felugró dialog ablakban nemmel válaszol, ekkor megszakad a betöltés művelet.</p>
Alternatív forgatókönyv	<p>1.A A felhasználó rossz típusú fájlt esetlegesen nem létező fájlt/útvonalat ad meg betöltésnél, ennek hatására a betöltés meghiúsul. majd elindul a hibakezelés</p> <p>Figyelem: az adatok betöltése esetén az éppen aktuális nyilvántartásban lévő adatok törlődnek.</p>

5. Adatok mentése:

Cím	Adatok mentése
Leírás	A felhasználó a nyilvántartásba vett adatokat/ szűrés eredményét kimenti az általa meghatározott helyre és névvel.
Aktorok	Felhasználó
Főforgatókönyv	<p>1. A felhasználó kiválasztja a mentés gombot, majd meghatározza a mentés eredménye képpen létrejövő fájl nevét, majd nyugtázza döntését melynek eredményeképpen létrejön a fájl a tárolt/szűrt adatokról.</p>
Alternatív forgatókönyv	1.A A mentés során valamely váratlan hiba bekövetkezése esetén megszakad a filementés és elindul a hibakezelés

6. Megtekintés:

Cím	Megtekintés
Leírás	A felhasználó megtekinti a nyilvántartásba vett adatokat/szűrés eredményét
Aktorok	Felhasználó
Főforgatókönyv	<p>1. A felhasználó megtudja tekinteni a táblázatban a szűrési feltételnek eleget tevő/betöltött állományba tartozó dokumentumokat és azok csatolt állományait, ha van. Illetve, ha már végzett bezárhatja az ablakot.</p>

7. Rendezés:

Cím	Rendezés
Leírás	A felhasználó a nyilvántartásba vett Dokumentumokat tudja rendezni attribútum alapján.
Aktorok	Felhasználó
Főforgatókönyv	<p>1. A felhasználó kiválasztja a rendezés alapját képező attribútum fejlécét, és ez növekvő, vagy csökkenő sorrendbe állítja az adatokat. Még egyszeri kiválasztás esetén az előző rendezés fordítottja lesz alkalmazva.</p>

Alternatív forgatókönyv	1.A.1 A felhasználó hibáján kívül bekövetkezett hiba elindítja a hibakezelést.
--------------------------------	---

8. Keresés:

Cím	Adat Szűrés
Leírás	A felhasználó a nyilvántartásba vett értékek között tud szűrni egy vagy több attribútum alapján.
Aktorok	Felhasználó
Főforgatókönyv	2. A felhasználó megadja a szűrési feltétel(ek)e)t és ez alapján elindítja a szűrést melynek eredménye a szűrési feltétel(ek)-nek megfelelő rekordok/dokumentumok.
Alternatív forgatókönyv	1.A.1 A felhasználó nem jól adja meg a szűrési feltétel(ek)e)-t és ennek eredménye, hogy a elindul a hibakezelés.

9. Fájl megnyitása:

Cím	Betöltés
Leírás	A felhasználó az előzetesen kimentett adatokat tudja betölteni a nyilvántartó rendszerbe, további használat céljából.
Aktorok	Felhasználó
Főforgatókönyv	1. A felhasználó kiválasztja a Fájlok/” Adatok Betöltése” menüpontot, meghatározza a betölteni kívánt fájlt, majd nyugtázza döntését ennek hatására az adatbázisba betöltődnek az előzetesen kimentett adatok.
Alternatív forgatókönyv	1.A A betöltés során nem létező fájl, hibás állomány, nem megfelelő formátum esete, megszakítja a fájlkezelést és elindítja a hibakezelést Minden adat betöltéskor az előzetesen tárolt adatok törlődnek a nyilvántartásból.

10. Hibakezelés:

Cím	Hibakezelés
Leírás	A felhasználó hibájából/ vagy azon kívül bekövetkezett hibák lekezelése és közlése a felhasználóval. Informatív jellegű kommunikáció a felhasználóval.
Aktorok	Felhasználó
Főforgatókönyv	1. Valamely program rossz futása, nem várt érték érkezése, szűrési vagy adatbeviteli szempont/megkötés megszegése elindítja a hibakezelést melynek eredménye a felhasználó számára, vagy error/info/dialog ablak mutatása.

5.Be- és Kimenetek, felhasznált forrástartalmak:

1.Be- és Kimenetek:

- A program használata során lehetőség van az egyes Nyilvántartásba vett dokumentumok/szűrési feltételek eredményének kimentésére és betöltésére.
- Ezek pontos formátuma: .ser típusú fájlok.
- Más típusú fájlok nem jelennek meg a fájlböngészőben betöltéskor. Illetőleg hibát jelez, ha mégis valami más kiterjesztésű fájlt szeretnénk importálni.
- Kimentéskor a név után meg kell adni a kiterjesztést is. Pld:
 - „test.ser” => .ser típusú fájl
 - „test” => kiterjesztés nélküli fájlt eredményez

2.Forrástartalmak:

- Az egyedüli külsőleges felhasznált forrástartalom egy .jar fájl, amely a Dátumválasztó megjelenítéséért és kezeléséért felelős az AddFrame-en.
- Ennek neve: „jdatepicker-1.3.4.jar”
- Ingyenesen letölthető .jar fájl/kiterjesztés a Dátumok grafikus kezelésére.
- Letölthető [innen](https://sourceforge.net/projects/jdatepicker/files/Releases/1.3.x/).¹

¹ <https://sourceforge.net/projects/jdatepicker/files/Releases/1.3.x/>