JEGYZŐKÖNYV

Modern Adatbázis Rendszerek
Féléves feladat

Áruház-raktár hálózat

Készítette: Stremler László
Neptunkód: AQYO8L

Dátum: 2025.04.03.

MISKOLC, 2025

Tartalomjegyzék

Bevezetés 3
1.feladat: 4
1.a feladat: Az adatbázis ER modell tervezése 4
1.b feladat: Az adatbázis konvertálása XDM modellre 5
1.c feladat: Az XDM modell alapján XML dokumentum készítése 5
1.d feladat: Az XML dokumentum alapján XMLSchema készítése – saját típusok, ref, key, keyref, speciális elemek 10
2.feladat: 16
2.a feladat: Adatolvasás 16
2.b feladat: Adatmódosítás 25
2.c feladat: Adatlekérdezés 28
2.d feladat: Adatírás 34
3. feladat 47

MISKOLC, 2025

Bevezetés

A feladat egy áruház-beszállító struktúrát mutat be. Többfajta kapcsolat jelenik meg a struktúrában: 1:1, 1:több, több:több kapcsolat. Az áruház és a beszállító is rendelkezik raktárral, ahol a termékek tárolva vannak. A raktárakban lévő termékek különböző tulajdonságokkal vannak ellátva (termék azonosító, név, darabszám, kategória). Amint egy termék bekerül az áruház raktárába, bővül pár tulajdonsággal (áruház azonosító, leírás, ár (több pénznemben is meg lehet adni az árat)). Megjelenítésre kerül az áruház-beszállító kapcsolat is, ahol egy változóban tárolva van az adott áruház heti átlagos termékberendelésének száma. A struktúra részletesebb bemutatása érdekében készült egy
ER modell is, amely tartalmazza az egyedeket, illetve az egyedek közötti kapcsolatokat.

Az ER-modell összesen 5 egyedet tartalmaz, melyek a következők:

- Áruház,
- Beszállító,
- Beszállító raktár,
- Áruház raktár tartalom,
- Akciós termékek

Először is az **Áruház** egyedet szeretném bemutatni. Ez az egyed tárolja az áruházak legfőbb tulajdonságait (*név, cím, áruház azonosító*). A **cím** egy összetett tulajdonságként jelenik meg, amely *az irányitószám, település, utca, házszám* elemekből épül fel. Ez az egyed az összeköttetés a beszállító és az áruházi raktár között. Elsődleges kulcs az AruhazID, amely minden áruház esetében egyedi és ez alapján lehet beazonosítani a raktárban, hogy az adott termék melyik áruház polcain található meg, illetve hogy az adott áruház mely beszállítóktól rendeli az áruit.

A következő fontos egyed a **Beszállító**. Ez az egyed tárolja a beszállítók adatait (*azonosító, név, termék kategória, átlagos kiszállítási idő*). A kategória egy előre meghatározott értéket felvevő egyed (*élelmiszer, üdítő, autó alkatrész, stb.*) Itt van meghatározva hogy mely beszállítók mely áruházaknak szállítanak. Mivel több áruház több beszállítótól is rendelhet árut és több beszállító is beszállíthat ugyanolyan kategóriájú árut az áruházba, ezért a beszállító és az áruház **között N:M (több:több) kapcsolat** van. A kapcsolatot pedig az **Áruház-beszállító kapcsolat** jellemzi, amely összeköti az Áruház és Beszállító egyedeket, valamint meghatározza az *átlagos rendelt árumennyiséget*.

Mivel a beszállító saját árukészlet nélkül nem ér sokat, ezért egy **beszállító raktár termék** egyedet is létrehoztam. Ebben a példában egy beszállító (mivel csak egy kategóriájú terméket

szállít) egy raktárral rendelkezik és egy raktárhoz csak egy beszállító tartozik, ezért ez a kettő között **1:1** kapcsolat található. Egy termék tulajdonságai a következők: *termék azonosító, név, darabszám, kategória.*

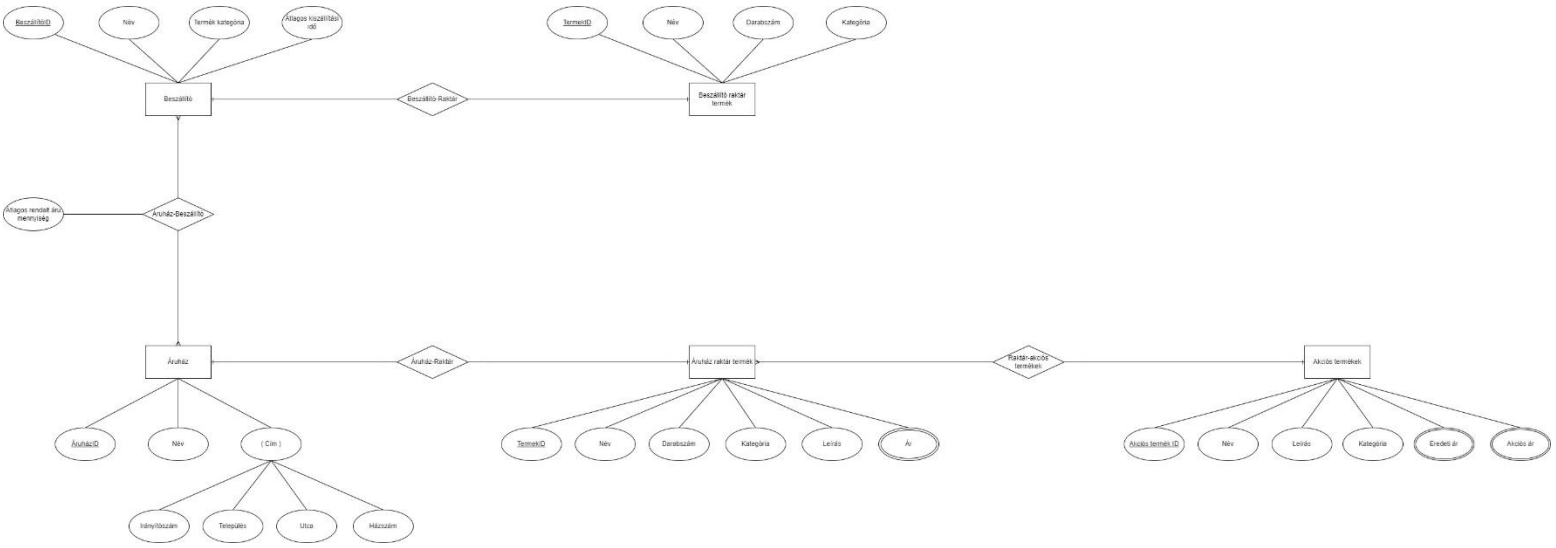
Az áruháznak is rendelkeznie kell saját raktárral, ahonnan feltölti az áruit és ahol tárolja a beérkezett árukat, így egy **áruház raktár** egyed is megtalálható a modellemben. Hasonlóképp, mint a beszállítónál, itt is egy áruház egy raktárral rendelkezik és egy raktár csak egy áruházat "szolgál ki", ezért a két egyed között **1:1** kapcsolat jellemző. Tulajdonságokat tekintetében magában foglalja a beszállító raktár tulajdonságait, illetve kibővítésre került a *kategória, leírás és ár* tulajdonságokkal. Az ár tulajdonság egy több értékű tulajdonság, ezáltal megadható hazai és külföldi valutában is az adott termék ára.

Végül pedig létrehoztam egy **akciós termékek** egyedet is, mivel az áruházak csak úgy tudnak gördülékenyen működni, ha néha engednek az árból. Az akciós termékek egyed közvetlen kapcsolatban áll az áruház raktárával, ezáltal könnyedén nyomonkövethető mely termékek akciósak. A két egyed között **1:N (egy:több)** kapcsolat található, mivel egy raktárbeli elemnek csak egy akciós variánsa lehet, viszont egy akciós termék több áruház raktárában is megjelenhet. Egy akciós termék a következő tulajdonságokkal van ellátva: *akciós termék azonosító, név, leírás, kategória, eredeti ár, akciós ár.* Ahogy az áruház raktárban, itt is az árra vonatkozó tulajdonságok többértékűek, ezáltal megadhatóak hazai és külföldi valutában is.

1.feladat:

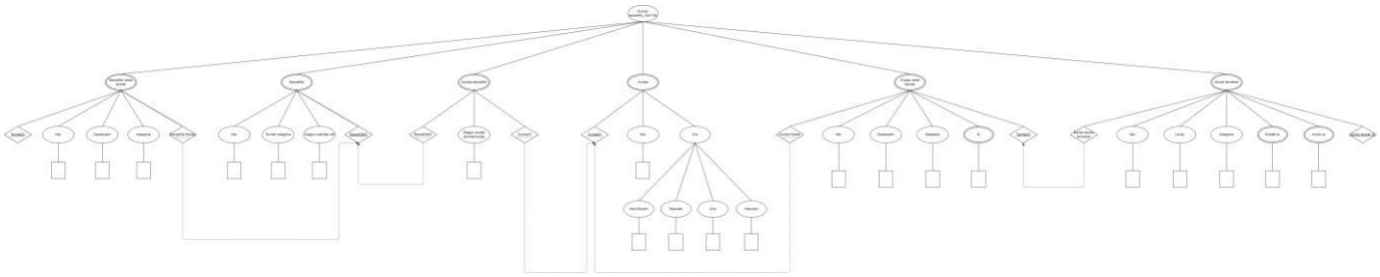
1.a feladat: Az adatbázis ER modell tervezése

Az ER-modellt a Draw.io nevezetű programban készítettem el. Először az egyedeket hoztam létre, amiket téglalappal jelöltem, majd azok tulajdonságait. amit elipszis alakzattal reprezentálok. Ezt követte az egyedek és tulajdonságaik összeköttetése, végül az egyedek összekötése a megfelelő kapcsolatot tükröző éllel. A többértékű tulajdonságnál dupla falú elipszist használtam. Törekedtem, hogy a feladat kiírásnak megfelelően érvényesítsem az összes kapcsolattípust, valamint betartsam a minimum egyedszámot.



1.b feladat: Az adatbázis konvertálása XDM modellre

Az ER modell elkészítése után létrehoztam az XDM modellt, amely reprezentálja az egyedek közötti kapcsolatokat egy mélyebb szinten. Itt már megjelennek az elsődleges, valamint idegenkulcsok. Az idegenkulcsok rámutatnak egy másik egyed elsődleges kulcsára, ezt nevezzük hivatkozásnak.



1.c feladat: Az XDM modell alapján XML dokumentum készítése

Az XDM modell elkészítése után elkezdtem felépíteni az XML dokumetumot, amely reprezentálja a modellt. Mivel minden elem többször is előfordulhat, ezért törekedtem – a feladat kiírásnak megfelelően – legalább 3 példányt készíteni mindegyikből. Az XML dokumentumban megjelenik struktúrált formában a fentebb említett cím tulajdonság, amely gyerekelemek segítségével írja le egy áruház pontos címét. Érdemes továbbá megemlíteni, hogy az akciós termékek elemben az árat tükröző Eredet_ar és Akcios_ar elemek mindig párban jelennek meg, tehát egy terméknek több pénznemben is megjelenik mind az akciós, mind az eredeti ára, ezzel szerettem volna tükrözni az elem többértékűségét. A pénznemet attribútumként lehet megadni az egyes ár elemekben.

```
<?xml version="1.0" encoding="UTF-8"?>
<Aruhaz-beszallito_AQYO8L xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
xs:noNamespaceSchemaLocation="XMLSchemaAQYO8L.xsd">
```

MISKOLC, 2025

```
<!-- Áruházak létrehozása és adatai megadása--> <Aruhazak>
    <!--Első
    áruház és adatai-->
    <Aruhaz aruhazid="1">
        <Nev>Tesco</Nev>
        <Cim>
            <Iranyitoszam>3544</Iranyitoszam>
            <Telepules>Miskolc</Telepules>
            <Utca>Karakter utca</Utca>
            <Hazszam>12</Hazszam>
        </Cim>
    </Aruhaz>

    <!--Második
    áruház és adatai-->
    <Aruhaz aruhazid="2">
        <Nev>Auchan</Nev>
        <Cim>
            <Iranyitoszam>1138</Iranyitoszam>
            <Telepules>Budapest</Telepules>
            <Utca>Lehel utca</Utca>
            <Hazszam>5</Hazszam>
        </Cim>
    </Aruhaz>

    <!--Harmadik
        áruház és adatai-->
      <Aruhaz aruhazid="3">
         <Nev>Spar</Nev>
        <Cim>
            <Iranyitoszam>6722</Iranyitoszam>
            <Telepules>Szeged</Telepules>
            <Utca>Arany János utca</Utca>
            <Hazszam>3</Hazszam>
        </Cim>
    </Aruhaz>
</Aruhazak>

<Beszallitok>
    <!-- Beszállítók létrehozása és adatai megadása--> <Beszallito
    beszallitoid="1">
        <Nev>Kerékkroncs KFT.</Nev>
        <Termek_kategoria>Gépjármű alkatrész</Termek_kategoria>
        <Atlagos_szallitasi_ido mertekegyseg="perc">48</Atlagos_szallitasi_ido> </Beszallito>

    <Beszallito beszallitoid="2">
        <Nev>Építőanyag KFT.</Nev>
```

```
            <Termek_kategoria>Építőanyag</Termek_kategoria>
            <Atlagos_szallitasi_ido mertekegyseg="ora">2</Atlagos_szallitasi_ido> </Beszallito>

        <Beszallito beszallitoid="3">
            <Nev>Elektronikai KFT.</Nev>
            <Termek_kategoria>Elektronika</Termek_kategoria>
            <Atlagos_szallitasi_ido mertekegyseg="ora">5</Atlagos_szallitasi_ido> </Beszallito>
</Beszallitok>

<!-- Áruház-Beszállító több:több kapcsolat megvalósítása-->

<Aruhaz-Beszallito aruhazid="1" beszallitoid="2">
        <Atlagos_Rendelt_Arumennyiseg>2</Atlagos_Rendelt_Arumennyiseg>
</Aruhaz-Beszallito>

<Aruhaz-Beszallito aruhazid="2" beszallitoid="3">
        <Atlagos_Rendelt_Arumennyiseg>15</Atlagos_Rendelt_Arumennyiseg>
</Aruhaz-Beszallito>

<Aruhaz-Beszallito aruhazid="3" beszallitoid="1">
        <Atlagos_Rendelt_Arumennyiseg>8</Atlagos_Rendelt_Arumennyiseg>
</Aruhaz-Beszallito>

<Raktarak>
        <!-- Áruház raktárak és adataik-->
        <Aruhaz_Raktar_Termek termekid="1" aruhazid="1">
            <Nev>Rántotthús</Nev>
            <Darabszam>2</Darabszam>
            <Kategoria>Élelmiszer</Kategoria>
            <Ar penznem="huf">1200</Ar>
        </Aruhaz_Raktar_Termek>
        <Aruhaz_Raktar_Termek termekid="2" aruhazid="1">
            <Nev>Tej</Nev>
            <Darabszam>10</Darabszam>
            <Kategoria>Élelmiszer</Kategoria>
            <Ar penznem="huf">300</Ar>
        </Aruhaz_Raktar_Termek>
        <Aruhaz_Raktar_Termek termekid="3" aruhazid="2">
            <Nev>Asztal</Nev>
            <Darabszam>5</Darabszam>
            <Kategoria>Bútor</Kategoria>
            <Ar penznem="huf">15000</Ar>
        </Aruhaz_Raktar_Termek>

        <!-- Beszállító raktárak és adataik-->
        <Beszallito_Raktar_Termek termekid="1" beszallitoid="1">
            <Nev>Autóalkatrész</Nev>
```

```
            <Darabszam>50</Darabszam>
            <Kategoria>Gépjármű alkatrész</Kategoria>
        </Beszallito_Raktar_Termek>
        <Beszallito_Raktar_Termek termekid="2" beszallitoid="2">
            <Nev>Csempe</Nev>
            <Darabszam>100</Darabszam>
            <Kategoria>Építőanyag</Kategoria>
        </Beszallito_Raktar_Termek>
        <Beszallito_Raktar_Termek termekid="3" beszallitoid="3">
            <Nev>Laptop</Nev>
            <Darabszam>20</Darabszam>
            <Kategoria>Elektronika</Kategoria>
        </Beszallito_Raktar_Termek>
    </Raktarak>

    <Akcios_Termekek>
        <!-- Akciós termékek és adataik-->
        <Akcios_Termek termekid="1" akciostermekid="1">
            <Nev>Farhát</Nev>
            <Leiras>Finom farhát</Leiras>
            <Kategoria>Élelmiszer</Kategoria>
            <Arak>
                <Eredeti_ar penznem="huf">1200.00</Eredeti_ar>
                <Eredeti_ar penznem="eu">4.00</Eredeti_ar> <Akcios_ar
                penznem="huf">1100.00</Akcios_ar> <Akcios_ar
                penznem="eu">3.75</Akcios_ar>
            </Arak>
        </Akcios_Termek>
        <Akcios_Termek termekid="2" akciostermekid="2">
            <Nev>Tej</Nev>
            <Leiras>Friss tej</Leiras>
            <Kategoria>Élelmiszer</Kategoria>
            <Arak>
                <Eredeti_ar penznem="huf">300.00</Eredeti_ar>
                <Eredeti_ar penznem="eu">1.00</Eredeti_ar> <Akcios_ar
                penznem="huf">250.00</Akcios_ar> <Akcios_ar
                penznem="eu">0.85</Akcios_ar>
            </Arak>
        </Akcios_Termek>
        <Akcios_Termek termekid="3" akciostermekid="3">
            <Nev>Asztal</Nev>
            <Leiras>Igényes asztal</Leiras>
            <Kategoria>Bútor</Kategoria>
            <Arak>
                <Eredeti_ar penznem="huf">15000.00</Eredeti_ar>
                <Eredeti_ar penznem="eu">50.00</Eredeti_ar> <Akcios_ar
                penznem="huf">12000.00</Akcios_ar> <Akcios_ar
                penznem="eu">40.8</Akcios_ar>
            </Arak>
```

```
        </Akcios_Termek>
        <Akcios_Termek termekid="4" akciostermekid="4">
            <Nev>Autóalkatrész</Nev>
            <Leiras>Minőségi alkatrész</Leiras>
            <Kategoria>Gépjármű alkatrész</Kategoria>
            <Arak>
                <Eredeti_ar penznem="huf">5000.0</Eredeti_ar>
                <Eredeti_ar penznem="eu">17.0</Eredeti_ar> <Akcios_ar
                penznem="huf">4500.0</Akcios_ar> <Akcios_ar
                penznem="eu">15.3</Akcios_ar>
            </Arak>
        </Akcios_Termek>
        <Akcios_Termek termekid="5" akciostermekid="5">
            <Nev>Csempe</Nev>
            <Leiras>Stílusos csempe</Leiras>
            <Kategoria>Építőanyag</Kategoria>
            <Arak>
                <Eredeti_ar penznem="huf">2000.0</Eredeti_ar>
                <Eredeti_ar penznem="eu">6.8</Eredeti_ar> <Akcios_ar
                penznem="huf">1800.0</Akcios_ar> <Akcios_ar
                penznem="eu">6.12</Akcios_ar>
            </Arak>
        </Akcios_Termek>
    </Akcios_Termekek>

</Aruhaz-beszallito_AQYO8L>
```

1.d feladat: Az XML dokumentum alapján XMLSchema készítése – saját típusok, ref, key, keyref, speciális elemek

Az XML dokumentumban szükség van megkötésekre, hogy létrejöjjön a teljes struktúra, ezért elkészítettem hozzá az XMLSchema-t, ami leírja az XML dokumentum felépítéséhez szükséges elemeket, valamint az elemek típusait. Továbbá lekorlátozza bizonyos elemek számosságát, ami szükséges az adott struktúrában. Definiálásra kerülnek benne az elsődleges, illetve idegenkulcsok is, amik összeköttetést biztosítanak az elemek között, ezáltal egyszerűbb lekérdezést lehetővé téve.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <!-- Saját általános típusok-->
    <xs:element name="Nev" type="xs:string" />
    <xs:element name="Termek_kategoria" type="kategoriaTipus" /> <xs:element
    name="Kategoria" type="kategoriaTipus" /> <xs:element name="Darabszam"
    type="xs:integer" />
```

```
<xs:element name="Atlagos_Rendelt_Arumennyiseg" type="xs:integer" /> <xs:element
name="Leiras" type="xs:string" />
<xs:attribute name="penznem" type="penznemTipus" />
<xs:attribute name="termekid" type="xs:integer" />
<xs:attribute name="beszallitoid" type="xs:integer" />
<xs:attribute name="aruhazid" type="xs:integer" />
<xs:attribute name="akciostermekid" type="xs:integer" />


<!-- Saját egyszerű típusok-->
<xs:simpleType name="mertekegysegTipus">
	<xs:restriction base="xs:string">
		<xs:enumeration value="perc" />
		<xs:enumeration value="ora" />
	</xs:restriction>
</xs:simpleType>

	<xs:simpleType name="kategoriaTipus">
			<xs:restriction base="xs:string">
		<xs:enumeration value="Gépjármű alkatrész" />
		<xs:enumeration value="Építőanyag" /> <xs:enumeration
		value="Elektronika" /> <xs:enumeration value="Élelmiszer"
		/> <xs:enumeration value="Bútor" /> <xs:enumeration
		value="Üdítő" /> <xs:enumeration value="Kávé" />
	</xs:restriction>
</xs:simpleType>

<xs:simpleType name="penznemTipus">
	<xs:restriction base="xs:string">
		<xs:enumeration value="huf" />
		<xs:enumeration value="eu" />
	</xs:restriction>
</xs:simpleType>


<!-- Saját komplex típusok-->
<xs:complexType name="Atlagos_szallitasi_ido_Tipus">
	<xs:simpleContent>
		<xs:extension base="xs:integer">
			<xs:attribute name="mertekegyseg" type="mertekegysegTipus" /> </xs:extension>
	</xs:simpleContent>
</xs:complexType>

<xs:complexType name="cimTipus">
	<xs:sequence>
		<xs:element name="Iranyitoszam" type="xs:integer" />
```

```
            <xs:element name="Telepules" type="xs:string" />
            <xs:element name="Utca" type="xs:string" />
            <xs:element name="Hazszam" type="xs:integer" />
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="EredetiAr_Tipus">
        <xs:simpleContent>
            <xs:extension base="xs:float">
                <xs:attribute name="penznem" type="penznemTipus" />
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>

    <xs:complexType name="AkciosAr_Tipus">
        <xs:simpleContent>
            <xs:extension base="xs:float">
                <xs:attribute name="penznem" type="penznemTipus" />
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>

    <xs:complexType name="Ar_Tipus">
        <xs:simpleContent>
            <xs:extension base="xs:float">
                <xs:attribute name="penznem" type="penznemTipus" />
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>

    <xs:complexType name="aruhazTipus">
        <xs:sequence>
            <xs:element ref="Nev" />
            <xs:element name="Cim" type="cimTipus" />
        </xs:sequence>
        <xs:attribute ref="aruhazid" />
    </xs:complexType>

    <xs:complexType name="aruhazBeszallitoTipus">
        <xs:sequence>
            <xs:element ref="Atlagos_Rendelt_Arumennyiseg" />
        </xs:sequence>
        <xs:attribute ref="aruhazid" />
        <xs:attribute ref="beszallitoid" />
    </xs:complexType>

    <xs:complexType name="ArakTipus">
        <xs:sequence>
                        <xs:element name="Eredeti_ar" type="EredetiAr_Tipus" maxOccurs="2" />
```

```
                <xs:element name="Akcios_ar" type="AkciosAr_Tipus" maxOccurs="2" />
            </xs:sequence>
        </xs:complexType>

        <xs:complexType name="beszallitoTipus">
            <xs:sequence>
                <xs:element ref="Nev" />
                <xs:element ref="Termek_kategoria" />
                <xs:element name="Atlagos_szallitasi_ido"
type="Atlagos_szallitasi_ido_Tipus" />
            </xs:sequence>
            <xs:attribute ref="beszallitoid" />
        </xs:complexType>

        <xs:complexType name="aruhazRaktarTermekTipus">
            <xs:sequence>
                <xs:element ref="Nev" />
                <xs:element ref="Darabszam" />
                <xs:element ref="Kategoria" />
                <xs:element name="Ar" type="Ar_Tipus" />
            </xs:sequence>
            <xs:attribute ref="termekid" />
            <xs:attribute ref="aruhazid" />
        </xs:complexType>

        <xs:complexType name="beszallitoRaktarTermekTipus">
            <xs:sequence>
                <xs:element ref="Nev" />
                <xs:element ref="Darabszam" />
                <xs:element ref="Kategoria" />
            </xs:sequence>
            <xs:attribute ref="termekid" />
            <xs:attribute ref="beszallitoid" />
        </xs:complexType>

        <xs:complexType name="akciosTermekTipus">
            <xs:sequence>
                <xs:element ref="Nev" />
                <xs:element ref="Leiras" />
                <xs:element ref="Kategoria" />
                <xs:element name="Arak" type="ArakTipus" />
            </xs:sequence>
            <xs:attribute ref="termekid" />
            <xs:attribute ref="akciostermekid" />
        </xs:complexType>

        <!--Közrefogó
        komplex típusok-->
```

MISKOLC, 2025

```
    <xs:complexType name="AruhazakTipus">
        <xs:sequence>
            <xs:element name="Aruhaz" type="aruhazTipus" minOccurs="3"
maxOccurs="unbounded">
            </xs:element>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="BeszallitokTipus">
        <xs:sequence>
            <xs:element name="Beszallito" type="beszallitoTipus" minOccurs="3"
maxOccurs="unbounded">
            </xs:element>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="RaktarakTipus">
        <xs:sequence>
            <xs:element name="Aruhaz_Raktar_Termek" type="aruhazRaktarTermekTipus"
minOccurs="3"
                maxOccurs="unbounded" />
            <xs:element name="Beszallito_Raktar_Termek"
type="beszallitoRaktarTermekTipus"
                minOccurs="3" maxOccurs="unbounded" />
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="Akcios_TermekekTipus">
        <xs:sequence>
            <xs:element name="Akcios_Termek" type="akciosTermekTipus" minOccurs="3"
                maxOccurs="unbounded" />
        </xs:sequence>
    </xs:complexType>

    <!-- Gyökérelem típusa-->
    <xs:complexType name="Aruhaz-beszallito_AQYO8LTipus">
        <xs:sequence>
            <xs:element name="Aruhazak" type="AruhazakTipus" />
            <xs:element name="Beszallitok" type="BeszallitokTipus" />
            <xs:element name="Aruhaz-Beszallito" type="aruhazBeszallitoTipus"
minOccurs="3"
                maxOccurs="unbounded" />
            <xs:element name="Raktarak" type="RaktarakTipus" />
            <xs:element name="Akcios_Termekek" type="Akcios_TermekekTipus" />
        </xs:sequence>
    </xs:complexType>

    <!--Gyökérelem
    definíciója-->
```

```
    <xs:element name="Aruhaz-beszallito_AQYO8L" type="Aruhaz-
beszallito_AQYO8LTipus">

        <!-- Elsődleges kulcsok-->
        <xs:key name="AruhazKulcs">
            <xs:selector xpath="Aruhazak/Aruhaz" />
            <xs:field xpath="@aruhazid" />
        </xs:key>

        <xs:key name="BeszallitoKulcs">
            <xs:selector xpath="Beszallitok/Beszallito" /> <xs:field
            xpath="@beszallitoid" />
        </xs:key>

        <xs:key name="BeszallitoTermekKulcs">
            <xs:selector xpath="Raktarak/Beszallito_Raktar_Termek" /> <xs:field
            xpath="@termekid" />
        </xs:key>

        <xs:key name="AruhazTermekKulcs">
            <xs:selector xpath="Raktarak/Aruhaz_Raktar_termek" /> <xs:field
            xpath="@termekid" />
        </xs:key>

        <xs:key name="AkciosTermekKulcs">
            <xs:selector xpath="Akcios_Termekek/Akcios_Termek" /> <xs:field
            xpath="@akciostermekid" />
        </xs:key>

        <!--Idegen
        kulcsok-->
            <xs:keyref name="Aruhaz-Raktar" refer="AruhazKulcs">
            <xs:selector xpath="Aruhaz_Raktar_Termek" /> <xs:field
            xpath="@aruhazid" />
        </xs:keyref>

        <xs:keyref name="AkciosTermek-Raktar" refer="AruhazTermekKulcs"> <xs:selector
            xpath="Akcios_Termek" />
            <xs:field xpath="@termekid" />
        </xs:keyref>

        <xs:keyref name="Beszallito-Raktar" refer="BeszallitoKulcs"> <xs:selector
            xpath="Beszallito_Raktar_Termek" /> <xs:field xpath="@beszallitoid" />
        </xs:keyref>

        <xs:keyref name="Aruhaz-BeszallitoAruhaz" refer="AruhazKulcs"> <xs:selector
            xpath="Aruhaz-Beszallito" />
            <xs:field xpath="@aruhazid" />
```

```
            </xs:keyref>

            <xs:keyref name="Aruhaz-BeszallitoBeszallito" refer="BeszallitoKulcs"> <xs:selector
                xpath="Aruhaz-Beszallito" />
                <xs:field xpath="@beszallitoid" />
            </xs:keyref>


            <!-- Egy-egy (1:1) kapcsolat kulcs-->
            <xs:unique    name="Beszallito-RaktarTermek_Kapcsolat">    <xs:selector
                xpath="Beszallito_Raktar_Termek"                />                <xs:field
                xpath="@beszallitoid" />
            </xs:unique>

            <xs:unique name="Aruhaz-RaktarTermek_Kapcsolat"> <xs:selector
                xpath="Aruhaz_Raktar_Termek"              />              <xs:field
                xpath="@aruhazid" />
            </xs:unique>

        </xs:element>


</xs:schema>
```

2.feladat:

2.a feladat: Adatolvasás

Az adatolvasó Java programhoz először beolvasom a dokumentumot. Ezután különböző függvényekkel kiírom struktúrált formában a konzolra. A struktúráltságért egyedi formázó függvényeket hoztam létre, amelyek bemenetként megkapják a beolvasott dokumentumot, valamint a kimeneti fájlt. Minden elemhez létrehoztam egy read függvényt, amely az adott elem összes elemét lekéri, megfelelő nevű változókba, majd ebből felépíti a struktúrát. A struktúra felépítésért a *printToFileAndConsole* függvény felel, amely bemenetként megkapja az elemet String-ként, valamint megkapja a konzol elérési útját (System.out) és a fájlt, amibe írni szeretnénk. Kimeneti fájl neve: *XMLAQYO8L_Copy.xml*

```
package hu.domparse.aqyo8l;

import java.io.File;
```

MISKOLC, 2025

```
import java.io.PrintStream;
import java.io.PrintWriter;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory; import
org.w3c.dom.Document; import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

public class DomReadAQYO8L {
	public static void main(String args[]) {
		try {
			// DocumentFactory inicializálása
					DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();

			// DocumentBuilder inicializálása
			DocumentBuilder builder = factory.newDocumentBuilder();

			File file = new File("XMLTaskAQYO8L\\1.feladat\\XMLAQYO8L.xml");

			// Dokumentum betöltése
			Document doc = builder.parse(file);
			doc.getDocumentElement().normalize();

			//Kimeneti fájl inicializálása PrintWriter outfile = new
			PrintWriter(new
File("XMLTaskAQYO8L\\2.feladat\\XMLAQYO8L_1.xml"), "UTF-8");

			// XML adatok kiírása
			printToFileAndConsole("<?xml version=\"1.0\" encoding=\"UTF-8\"?>", System.out,
outfile);
			printToFileAndConsole(
					"<Aruhaz-beszallito_AQYO8L
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"XMLSchemaAQYO8L.xsd\">",
					System.out, outfile);

			//	Áruházak beolvasása
			readAruhaz(doc, outfile);

			//	Beszállítók beolvasása
			readBeszallito(doc, outfile);

			//	Áruház-Beszállító kapcsolatok beolvasása
			readAruhazBeszallito(doc, outfile);

			//	Raktárak beolvasása
			readRaktarak(doc, outfile);
```

MISKOLC, 2025

```
                //    Akciós termékek beolvasása
                readAkciosTermekek(doc, outfile);

                //    XML gyökérelem lezárása
                printToFileAndConsole("</Aruhaz-beszallito_AQYO8L>", System.out,
outfile);

                //    Kimeneti fájl lezárása
                outfile.close();

            } catch (Exception e) {
                e.printStackTrace();
            }
        }

        // Kiíró metódus
        private static void printToFileAndConsole(final String msg, PrintStream console, PrintWriter
file) {
            console.println(msg);
            file.println(msg);
        }

        // Eleme kiírás formázó metódus
        private static void printElement(String elementName, String content, PrintWriter file) {
            printToFileAndConsole("                                <" + elementName + ">" + content + "</"
+  elementName + ">", System.out, file);
        }

        // Cím kiírás formázó metódus
        private static void printCim(Element cimElement, PrintWriter file) {
            printToFileAndConsole("                                        <" + cimElement.getNodeName() + ">",
System.out, file);
            printToFileAndConsole(
                    "                        <" +
cimElement.getElementsByTagName("Iranyitoszam").item(0).getNodeName() +">"+
cimElement.getElementsByTagName("Iranyitoszam").item(0).getTextContent() + "</"
                            +
cimElement.getElementsByTagName("Iranyitoszam").item(0).getNodeName() + ">",

            printToFileAndConsole(
                    "                        <" +
cimElement.getElementsByTagName("Telepules").item(0).getNodeName()
                            +">"+
cimElement.getElementsByTagName("Telepules").item(0).getTextContent() + "</"
                            +
cimElement.getElementsByTagName("Telepules").item(0).getNodeName() + ">",
```

```
                        System.out, file);
                printToFileAndConsole(
                        "                           <"   +
cimElement.getElementsByTagName("Utca").item(0).getNodeName() + ">"
                                +
cimElement.getElementsByTagName("Utca").item(0).getTextContent() + "</"
                                +
cimElement.getElementsByTagName("Utca").item(0).getNodeName() + ">", System.out,
                        file);
                printToFileAndConsole("                             <" +
cimElement.getElementsByTagName("Hazszam").item(0).getNodeName()
                        +">"+
cimElement.getElementsByTagName("Hazszam").item(0).getTextContent() + "</"
                        +  cimElement.getElementsByTagName("Hazszam").item(0).getNodeName()
+  ">", System.out, file);
                printToFileAndConsole("                         </" + cimElement.getNodeName() +
                ">", System.out, file);
        }

    // Áruházakat beolvasó metódus
    private static void readAruhaz(Document document, PrintWriter file) { NodeList aruhazList =
            document.getElementsByTagName("Aruhaz");
            printToFileAndConsole(" <Aruhazak>", System.out, file); for (int temp = 0;
            temp < aruhazList.getLength(); temp++) {
                Node node = aruhazList.item(temp);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element aruhazElement = (Element) node; String nev =
aruhazElement.getElementsByTagName("Nev").item(0).getTextContent(); String aruhazId =
                    aruhazElement.getAttribute("aruhazid"); Element cimElement = (Element)
aruhazElement.getElementsByTagName("Cim").item(0);

                    printToFileAndConsole(" <Aruhaz aruhazid=\"" + aruhazId + "\">", System.out, file);
                    printElement("Nev", nev, file);
                    printCim(cimElement, file);
                    printToFileAndConsole("                     </Aruhaz>", System.out, file);
                }
            }
            printToFileAndConsole("             </Aruhazak>", System.out, file);
    }

    // Beszállítókat beolvasó metódus
    private static void readBeszallito(Document document, PrintWriter file) { NodeList beszallitoList =
            document.getElementsByTagName("Beszallito");
            printToFileAndConsole(" <Beszallitok>", System.out, file); for (int temp = 0; temp <
            beszallitoList.getLength(); temp++) {
                Node node = beszallitoList.item(temp);
```

```
                if (node.getNodeType() == Node.ELEMENT_NODE) { Element
                    beszallitoElement = (Element) node; String nev =
beszallitoElement.getElementsByTagName("Nev").item(0).getTextContent(); String beszallitoid
                    =
beszallitoElement.getAttribute("beszallitoid"); String
                    termekKategoria =
            beszallitoElement.getElementsByTagName("Termek_kategoria").item(0)
                                .getTextContent();
                    Element atlagosSzallitasiIdoElement = (Element) beszallitoElement
                            .getElementsByTagName("Atlagos_szallitasi_ido").item(0); String
                    atlagosSzallitasiIdoString =
atlagosSzallitasiIdoElement.getTextContent();
                    String          atlagosSzallitasiIdoMertekegyseg          =
atlagosSzallitasiIdoElement.getAttribute("mertekegyseg");

                    printToFileAndConsole("                    <Beszallito beszallitoid=\"" +
beszallitoid + "\">", System.out, file);
                    printElement("Nev", nev, file);
                    printElement("Termek_kategoria", termekKategoria, file);
                    printToFileAndConsole(
                            " <Atlagos_szallitasi_ido mertekegyseg=\"" +
atlagosSzallitasiIdoMertekegyseg
                                    + "\">" + atlagosSzallitasiIdoString +
"</Atlagos_szallitasi_ido>",
                            System.out, file);
                    printToFileAndConsole("                    </Beszallito>", System.out, file);
                }

                printToFileAndConsole("                </Beszallitok>", System.out, file);
            }
        }

        // Áruház-Beszállító kapcsolatokat beolvasó metódus
        private static void readAruhazBeszallito(Document document, PrintWriter file) { NodeList
            aruhazBeszallitoList = document.getElementsByTagName("Aruhaz-
Beszallito");
            for (int temp = 0; temp < aruhazBeszallitoList.getLength(); temp++) { Node node =
                aruhazBeszallitoList.item(temp);
                if (node.getNodeType() == Node.ELEMENT_NODE) { Element
                    aruhazBeszallitoElement = (Element) node;
                    String atlagos_Rendelt_Arumennyiseg = aruhazBeszallitoElement
                                    .getElementsByTagName("Atlagos_Rendelt_Arumennyiseg").item(
0).getTextContent();
                            String beszallitoid =
aruhazBeszallitoElement.getAttribute("beszallitoid");
                                    String aruhazid = aruhazBeszallitoElement.getAttribute("aruhazid");
```

MISKOLC, 2025

```
                    printToFileAndConsole(" <Aruhaz-Beszallito aruhazid=\"" + aruhazid + "\"
beszallitoid=\""
                            +  beszallitoid + "\">", System.out, file);
                    printToFileAndConsole("<Atlagos_Rendelt_Arumennyiseg>"+at
lagos_Rendelt_Arumennyiseg+"</Atlagos_Rendelt_Arumennyiseg>", System.out, file);
                    printToFileAndConsole("                </Aruhaz-Beszallito>", System.out,
file);
                }
            }
        }

        // Raktárakat beolvasó metódus
        private static void readRaktarak(Document document, PrintWriter file) { NodeList
            aruhazRaktarTermekList =
document.getElementsByTagName("Aruhaz_Raktar_Termek"); NodeList
            beszallitoRaktarTermekList =
document.getElementsByTagName("Beszallito_Raktar_Termek");
            printToFileAndConsole(" <Raktarak>", System.out, file);
            for (int temp = 0; temp < aruhazRaktarTermekList.getLength(); temp++) { Node node =
                aruhazRaktarTermekList.item(temp);
                if (node.getNodeType() == Node.ELEMENT_NODE) { Element
                    aruhaRaktarElement = (Element) node;
                    String termekid = aruhaRaktarElement.getAttribute("termekid");
                    String aruhazid = aruhaRaktarElement.getAttribute("aruhazid");
                    String nev =
aruhaRaktarElement.getElementsByTagName("Nev").item(0).getTextContent(); String darabszam
                    =
aruhaRaktarElement.getElementsByTagName("Darabszam").item(0).getTextContent(); String kategoria =
aruhaRaktarElement.getElementsByTagName("Kategoria").item(0).getTextContent(); Element arElement =
                    (Element)
aruhaRaktarElement.getElementsByTagName("Ar").item(0); String ar =
                    arElement.getTextContent();
                        String penznem = arElement.getAttribute("penznem");

                    printToFileAndConsole(" <Aruhaz_Raktar_Termek aruhazid=\"" + aruhazid + "\"
termekid=\""
                            +  termekid + "\">", System.out, file);
                    printElement("Nev", nev, file); printElement("Darabszam",
                    darabszam, file); printElement("Kategoria", kategoria, file);
                    printToFileAndConsole("                        <Ar penznem=\"" + penznem +
"\">" + ar + "</Ar>", System.out, file);
                    printToFileAndConsole("                    </Aruhaz_Raktar_Termek>", System.out,
file);
                }

                for (int temp2 = 0; temp < beszallitoRaktarTermekList.getLength();
temp++) {
```

```
                    Node node2 = beszallitoRaktarTermekList.item(temp2); if
                    (node2.getNodeType() == Node.ELEMENT_NODE) {
                        Element beszallitoRaktarElement = (Element) node2; String
                        termekid =
beszallitoRaktarElement.getAttribute("termekid"); String
                        beszallitoid =
            beszallitoRaktarElement.getAttribute("beszallitoid");
                        String nev =
beszallitoRaktarElement.getElementsByTagName("Nev").item(0).getTextContent();
                        String darabszam =
beszallitoRaktarElement.getElementsByTagName("Darabszam").item(0)
                                .getTextContent();
                        String kategoria =
beszallitoRaktarElement.getElementsByTagName("Kategoria").item(0)
                                .getTextContent();

                        printToFileAndConsole(
                                "                <Beszallito_Raktar_Termek beszallitoid=\"" +
beszallitoid + "\" termekid=\""
                                        +  termekid + "\">",
                                System.out, file);
                        printElement("Nev", nev, file);
                        printElement("Darabszam", darabszam, file);
                        printElement("Kategoria", kategoria, file);
                        printToFileAndConsole("                    </Beszallito_Raktar_Termek>",
System.out, file);
                    }
                }
                    printToFileAndConsole("            </Raktarak>", System.out, file);
            }
        }

        // Akciós termékeket beolvasó metódus
        private static void readAkciosTermekek(Document document, PrintWriter file) { NodeList
            akciosTermekekList =
document.getElementsByTagName("Akcios_Termek");
            printToFileAndConsole("            <Akcios_Termekek>", System.out, file);
            for (int temp = 0; temp < akciosTermekekList.getLength(); temp++) { Node node =
                akciosTermekekList.item(temp);
                if (node.getNodeType() == Node.ELEMENT_NODE) { Element
                    akciosTermekElement = (Element) node;
                    String termekid = akciosTermekElement.getAttribute("termekid"); String
                    akciostermekid =
akciosTermekElement.getAttribute("akciostermekid"); String nev =
akciosTermekElement.getElementsByTagName("Nev").item(0).getTextContent(); String leiras =
akciosTermekElement.getElementsByTagName("Leiras").item(0).getTextContent();
```

```
                    String kategoria =
akciosTermekElement.getElementsByTagName("Kategoria").item(0).getTextContent();

                        NodeList eredetiArList =
        akciosTermekElement.getElementsByTagName("Eredeti_ar");
                        NodeList akciosArList =
        akciosTermekElement.getElementsByTagName("Akcios_ar");
```

```
                    printToFileAndConsole("                    <Akcios_Termek termekid=\"" +
termekid + "\" akciostermekid=\""
                            +  akciostermekid + "\">", System.out, file);
                    printElement("Nev", nev, file); printElement("Leiras", leiras, file);
                    printElement("Kategoria", kategoria, file);

                    printToFileAndConsole("                        <Arak>", System.out, file);

                    for (int temp2 = 0; temp2 < eredetiArList.getLength(); temp2++) { Node node2 =
                        eredetiArList.item(temp2);
                        if (node2.getNodeType() == Node.ELEMENT_NODE) {
                            Element eredetiAr = (Element) node2;
                            String penznem = eredetiAr.getAttribute("penznem"); String ar =
                            eredetiAr.getTextContent(); printToFileAndConsole(
                                    "                            <Eredeti_ar penznem=\"" + penznem
+ "\">" + ar + "</Eredeti_ar>",
                                        System.out, file);
                        }
                    }

                    for (int temp3 = 0; temp3 < akciosArList.getLength(); temp3++) { Node node3 =
                        akciosArList.item(temp3);
                        if (node3.getNodeType() == Node.ELEMENT_NODE) {
                            Element akciosAr = (Element) node3;
                            String penznem = akciosAr.getAttribute("penznem"); String ar =
                            akciosAr.getTextContent(); printToFileAndConsole(
                                    "                            <Akcios_ar penznem=\"" + penznem +
"\">" + ar + "</Akcios_ar>",
                                        System.out, file);
                        }
                    }

                    printToFileAndConsole("                        </Arak>", System.out, file);

                     printToFileAndConsole("                            </Akcios_Termek>", System.out, file);
                }
            }
            printToFileAndConsole("        </Akcios_Termekek>", System.out, file);
```

```
    }
}
```

2.b feladat: Adatmódosítás

Az adatmódosító függvényben először beolvasásra kerül az első feladat XML dokumentuma, majd annak elemei kerülnek módosításra. Módosításra kerülnek az áruházak és beszállítók nevei, valamint az áruház termékek árainak pénzneme. Ez összesen 9 db módosítás. A módosított dokumentumot végül kiírja a konzolra.

```
package hu.domparse.aqyo8l;

import java.io.IOException;
import java.io.StringWriter;
import java.util.Random;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory; import
javax.xml.parsers.ParserConfigurationException; import
javax.xml.transform.OutputKeys; import
javax.xml.transform.Transformer;
import javax.xml.transform.TransformerException; import
javax.xml.transform.TransformerFactory; import
javax.xml.transform.dom.DOMSource; import
javax.xml.transform.stream.StreamResult;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

public class DomModifyAQYO8L {
	public static void main(String args[]) {
		try {
			// DocumentBuilder inicializálása
			DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance(); DocumentBuilder
			builder = factory.newDocumentBuilder();
			//	Dokumentum beolvasása
			Document document =
builder.parse("XMLTaskAQYO8L\\1.feladat\\XMLAQYO8L.xml");

			//	Dokumentum módosítása
			modifyNodes(document);

			//	Dokumentum kiírása a konzolra a módosítás után
			printXML(document);
```

```
        } catch (ParserConfigurationException | SAXException | IOException e) { e.printStackTrace();
        }
    }

    private static void printXML(Document document) { try {
            //  TransformerFactory és Transformer osztályok példányosítása
            TransformerFactory transformerFactory =
TransformerFactory.newInstance();
            Transformer transformer = transformerFactory.newTransformer();

            // Behúzás beállítása a transformerben
            transformer.setOutputProperty(OutputKeys.INDENT, "yes");

            //  StringWriter osztály példányosítása, amiben eltároljuk a
dokumentumot
            StringWriter stringWriter = new StringWriter();

            //  Dokumentum string-gé alakítása
            transformer.transform(new DOMSource(document), new
StreamResult(stringWriter));

            // Dokumentum kiírása a konzolra
            System.out.println(stringWriter.toString());
        } catch (TransformerException e) {
            e.printStackTrace();
        }
    }

    private static void modifyNodes(Document document) { // Lekéri az
        összes Aruhaz node-ot
            NodeList aruhazNodeList = document.getElementsByTagName("Aruhaz");

        // Végigiterál a node-okon
        for (int i = 0; i < aruhazNodeList.getLength(); i++) { Node node =
            aruhazNodeList.item(i);

            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element element = (Element) node;

                // Lekéri a Nev node-ot
                        Node aruhazNevNode = element.getElementsByTagName("Nev").item(0);

                // Módosítja a node értékét
                    aruhazNevNode.setTextContent("Aruhaz" + (i + 1));
            }
        }
```

MISKOLC, 2025

```
            //    Lekéri az összes ÁruházRaktárTermék node-ot NodeList
            aruhazRaktarTermekNodeList =
document.getElementsByTagName("Aruhaz_Raktar_Termek");

            // Végigiterál a node-okon
            for (int i = 0; i < aruhazRaktarTermekNodeList.getLength(); i++) { Node node =
                  aruhazRaktarTermekNodeList.item(i);

                  if (node.getNodeType() == Node.ELEMENT_NODE) {
                        Element element = (Element) node;

                        String newPenznem = "";

                        Random random = new Random();

                        int randomErtek = random.nextInt(2);

                        switch (randomErtek) {
                              case 1:
                                    newPenznem = "EUR";
                                    break;

                              case 2:
                                    newPenznem = "USD";
                                    break;

                              default:
                                    newPenznem = "CAD";
                                    break;
                        }

                        // Lekéri a Nev node-ot
                        Node arNode = element.getElementsByTagName("Ar").item(0);

                        // Node Element-té alakítása
                        Element arElement = (Element) arNode;

                        // Módosítja az attribútum értékét
                        arElement.setAttribute("penznem", newPenznem);
                  }
            }

              // Lekéri az összes Beszallito node-ot
            NodeList beszallitoNodeList = document.getElementsByTagName("Beszallito");

            // Végigiterál a node-okon
            for (int i = 0; i < beszallitoNodeList.getLength(); i++) { Node node =
                  beszallitoNodeList.item(i);
```

```
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element element = (Element) node;

                    //    Lekéri a Nev node-ot Node
                    beszallitoNevNode =
element.getElementsByTagName("Nev").item(0);

                    // Módosítja a node értékét
                    beszallitoNevNode.setTextContent("Beszallito" + (i + 1));
                }
            }

        }
}
```

2.c feladat: Adatlekérdezés

Ebben a feladatban próbáltam egyszerű és összetett lekérdezéseket is készíteni. Az első lekérdezés lekéri az összes áruházat.

A második lekérdezésben lekérdezésre kerülnek az áruház-beszállító kapcsolatok annyi csavarral, hogy nem az ID-kat íratom ki, hanem az áruház és a beszállító nevét, ezáltal több elemcsoporton átívelő lekérdezés jön létre.

A harmadik lekérdezésben implementáltam egy maximum keresést az átlagos rendelt árumennyiség tekintetében. Végigiterál az összes áruház-beszállító kapcsolaton és megkeresi a legtöbb árut rendelt áruházat, majd ennek a nevét írja ki konzolra. (Tehát itt is több elemcsoporton átívelő lekérdezés történik).

A negyedik lekérdezésben az összes Miskolcon található áruházat kérdezem le. Ezt úgy valósítottam meg, hogy az összes áruházon végigiterálok és lekérem a címeiket. Ahol a település gyerekelem megegyezik Miskolccal, azt az áruházat kiíratom a konzolra.

Az utolsó lekérdezésben pedig a maximum 2 óra kiszállító beszállítókat kérdezem le. Végigiterálok az összes beszállítón és ahol a kiszállítási idő 2 óránál vagy 48 percnél nem nagyobb, azt kiíratom a konzolra.

```
package hu.domparse.aqyo8l;

import java.io.IOException;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory; import
javax.xml.parsers.ParserConfigurationException;

import org.w3c.dom.Document;
import org.xml.sax.SAXException;
import org.w3c.dom.NodeList;
```

```
import org.w3c.dom.Node;
import org.w3c.dom.Element;

public class DomQueryAQYO8L {
	public static void main(String args[]) {
		try {
			// DocumentBuilder inicializálása
			DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance(); DocumentBuilder
			builder = factory.newDocumentBuilder();
			//	Dokumentum beolvasása
			Document document =
builder.parse("XMLTaskAQYO8L\\1.feladat\\XMLAQYO8L.xml");

			//	Üres sor beszúrása a konzolon, a jobb olvashatóság érdekében
			System.out.println();

			System.out.println("Összes áruház lekérdezése:"); // Összes áruház
			lekérdezése
			NodeList aruhazList = document.getElementsByTagName("Aruhaz"); // Végigiterál az
			összes Aruhaz Node-on
			for (int i = 0; i < aruhazList.getLength(); i++) { Node node =
				aruhazList.item(i);
				// Megnézi, hogy az elem elem típusú-e
				if (node.getNodeType() == Node.ELEMENT_NODE) {
					Element aruhaz = (Element) node;

					//	Kiírja az áruház ID-ját és nevét
					System.out.println("Áruház ID: " +
aruhaz.getAttribute("aruhazid") + " Áruház neve: "
							+
aruhaz.getElementsByTagName("Nev").item(0).getTextContent());
				}
			}

			//	Üres sor beszúrása a konzolon, a jobb olvashatóság érdekében
			System.out.println();

			System.out.println("Áruház-Beszállító kapcsolatok lekérdezése:");

			//	Áruház-Beszállító kapcsolatok lekérdezése
			NodeList aruhazBeszallitoKapcsolatList =
document.getElementsByTagName("Aruhaz-Beszallito");

			// Összes áruház lekérdezése
			aruhazList = document.getElementsByTagName("Aruhaz");

			// Összes beszállító lekérdezése
				NodeList beszallitoList = document.getElementsByTagName("Beszallito");
```

MISKOLC, 2025

```
                // Végigiterál az összes Aruhaz-Beszallito Node-on
                for (int i = 0; i < aruhazBeszallitoKapcsolatList.getLength(); i++) { Node node =
                        aruhazBeszallitoKapcsolatList.item(i);

                        //    Inicializál egy stringet, ami a kiírandó sor lesz
                        String kiirtSor = "";

                        //    Megnézi, hogy az elem elem típusú-e
                        if (node.getNodeType() == Node.ELEMENT_NODE) {
                                Element aruhazBeszallitoKapcsolat = (Element) node;

                                // Végigiterál az összes áruházon
                                for (int j = 0; j < aruhazList.getLength(); j++) { Node aruhazNode =
                                        aruhazList.item(j);

                                        // Megnézi, hogy az elem elem típusú-e
                                        if (aruhazNode.getNodeType() == Node.ELEMENT_NODE) {
                                                Element aruhaz = (Element) aruhazNode;

                                                // Megnézi, hogy az áruház ID-ja megegyezik-e az áruház
ID-jával az
                                                //    áruház-beszállító kapcsolatban if
                                                (aruhaz.getAttribute("aruhazid")
                                                                .equals(aruhazBeszallitoKapcsolat.getAttribute(
"aruhazid"))) {
                                                        //    Hozzáadja a kiírandó sorhoz az áruház nevét kiirtSor +=
                                                        "Áruház neve: "
                                                                        +
aruhaz.getElementsByTagName("Nev").item(0).getTextContent() + " ";
                                                }
                                        }
                                }

                                // Végigiterál az összes beszállítón
                                for (int j = 0; j < beszallitoList.getLength(); j++) { Node beszallitoNode =
                                        beszallitoList.item(j);
                                        // Megnézi, hogy az elem elem típusú-e
                                        if (beszallitoNode.getNodeType() == Node.ELEMENT_NODE) { Element
                                                beszallito = (Element) beszallitoNode;

                                                // Megnézi, hogy a beszállító ID-ja megegyezik-e a
beszállító ID-jával az
                                                // áruház-beszállító kapcsolatban
                                                if (beszallito.getAttribute("beszallitoid")
                                                                .equals(aruhazBeszallitoKapcsolat.getAttribute(
"beszallitoid"))) {
                                                        //    Hozzáadja a kiírandó sorhoz a beszállító nevét kiirtSor +=
                                                        "Beszállító neve: "
```

```
                                                +
beszallito.getElementsByTagName("Nev").item(0).getTextContent();
                                        }
                                    }
                                }

                                //    Kiírja a kapcsolatban álló áruházak és beszállítók neveit
                                System.out.println(kiirtSor);
                            }
                        }

                        //    Üres sor beszúrása a konzolon, a jobb olvashatóság érdekében
                        System.out.println();

                        System.out.println("Legtöbb árut rendelő áruház lekérdezése: ");

                        //    Áruház-Beszállító kapcsolatok lekérdezése aruhazBeszallitoKapcsolatList =
                        document.getElementsByTagName("Aruhaz-
Beszallito");
```

```
                        // Összes áruház lekérdezése
                        aruhazList = document.getElementsByTagName("Aruhaz");

                        //    Inicializál egy stringet, ami a kiírandó sor lesz
                        String kiirtSor = "";

                        //    Inicializál egy int-et, ami a darabszámot fogja tárolni int aruDarab = 0;

                        //    Végigiterál az összes Aruhaz-Beszallito Node-on
                        for (int i = 0; i < aruhazBeszallitoKapcsolatList.getLength(); i++) { Node node =
                            aruhazBeszallitoKapcsolatList.item(i);

                            // Megnézi, hogy az elem elem típusú-e
                            if (node.getNodeType() == Node.ELEMENT_NODE) {
                                Element aruhazBeszallitoKapcsolat = (Element) node;

                                // Végigiterál az összes áruházon
                                for (int j = 0; j < aruhazList.getLength(); j++) { Node aruhazNode =
                                    aruhazList.item(j);

                                    // Megnézi, hogy az elem elem típusú-e
                                    if (aruhazNode.getNodeType() == Node.ELEMENT_NODE) {
                                        Element aruhaz = (Element) aruhazNode;

                                        int rendeltMennyiseg =
Integer.valueOf(aruhazBeszallitoKapcsolat
                                                .getElementsByTagName("Atlagos_Rendelt_Arumenny
iseg").item(0).getTextContent());
```

```
                                    // Megnézi, hogy az áruház ID-ja megegyezik-e az áruház
ID-jával az
                                    //     áruház-beszállító kapcsolatban if
                                    (aruhaz.getAttribute("aruhazid")
                                            .equals(aruhazBeszallitoKapcsolat.getAttribute(
"aruhazid"))
                                            &&rendeltMennyiseg > aruDarab) {
                                        //     Beállítja a legtöbb árut rendelő áruház
darabszámát
                                        aruDarab = rendeltMennyiseg;
                                        //     Hozzáadja a kiírandó sorhoz az áruház nevét kiirtSor =
                                        "Áruház neve: " +
aruhaz.getElementsByTagName("Nev").item(0).getTextContent()
                                                + ", rendelt mennyiség: " +
rendeltMennyiseg;
                                    }
                                }
                            }
                        }
                    }
                    // Kiírja a legtöbb árut rendelő áruház nevét és a rendelt áruk
darabszámát
                    System.out.println(kiirtSor);

                    //    Üres sor beszúrása a konzolon, a jobb olvashatóság érdekében
                    System.out.println();

                    System.out.println("Összes Miskolcon található áruház lekérdezése:"); // Összes áruház
                    lekérdezése
                    aruhazList = document.getElementsByTagName("Aruhaz"); // Végigiterál
                    az összes Aruhaz Node-on
                    for (int i = 0; i < aruhazList.getLength(); i++) { Node node =
                        aruhazList.item(i);
                        // Megnézi, hogy az elem elem típusú-e
                        if (node.getNodeType() == Node.ELEMENT_NODE) {
                            Element aruhaz = (Element) node;
                            // Lekéri a címet
                            Node cimNode = aruhaz.getElementsByTagName("Cim").item(0); Element
                            cim = (Element) cimNode;
                            //     Lekéri a települést
                            Node telepulesNode =
cim.getElementsByTagName("Telepules").item(0);

                                                    // Ha az áruház Miskolcon található, akkor kiírja az áruház ID-
ját, nevét és
                            // pontos címét
                            if (telepulesNode.getTextContent().equals("Miskolc")) {
```

```
                                System.out.println("Áruház ID: " +
aruhaz.getAttribute("aruhazid") + " Áruház neve: "
                                        +
aruhaz.getElementsByTagName("Nev").item(0).getTextContent() + " Címe: "
                                        +
cim.getElementsByTagName("Iranyitoszam").item(0).getTextContent() + ", "
                                        +
cim.getElementsByTagName("Telepules").item(0).getTextContent() + ", "
                                        +
            cim.getElementsByTagName("Utca").item(0).getTextContent() + ", "
                                        +
cim.getElementsByTagName("Hazszam").item(0).getTextContent());
                                }
                            }
                        }

                        //    Üres sor beszúrása a konzolon, a jobb olvashatóság érdekében
                        System.out.println();

                        System.out.println("Olyan beszállítók lekérdezése, amelyek legfeljebb 2 óra alatt szállítanak
ki:");
                        // Összes beszallito lekérdezése
                        beszallitoList = document.getElementsByTagName("Beszallito");

                        //    Inicializál egy stringet, ami a kiírandó sor lesz kiirtSor = "";

                        //    Végigiterál az összes Beszallito Node-on
                        for (int i = 0; i < beszallitoList.getLength(); i++) { Node node =
                            beszallitoList.item(i);

                            // Megnézi, hogy az elem elem típusú-e
                            if (node.getNodeType() == Node.ELEMENT_NODE) {
                                Element beszallito = (Element) node;

                                //    Lekéri az átlagos szállítási időt
                                Node szallitasiIdoNode =
beszallito.getElementsByTagName("Atlagos_szallitasi_ido").item(0);

                                // Átcastolja Element típusra
                                Element szallitasiIdo = (Element) szallitasiIdoNode;

                                //    Lekéri az időtartam mértékegységét
                                String mertekegyseg =
szallitasiIdo.getAttribute("mertekegyseg");

                                //    Ha az átlagos szállítási idő legfeljebb 2 óra, akkor
hozzáadja a kiírandó
                                //    sorhoz
```

```
                        if (mertekegyseg.equals("ora") &&
Integer.valueOf(szallitasiIdo.getTextContent()) <= 2
                                || mertekegyseg.equals("perc") &&
Integer.valueOf(szallitasiIdo.getTextContent()) <= 120) {
                                kiirtSor += "Beszállító ID: " +
            beszallito.getAttribute("beszallitoid") + ", Beszállító neve: "
                                +
beszallito.getElementsByTagName("Nev").item(0).getTextContent() + ", Szállítási idő: "
                                + szallitasiIdo.getTextContent() + " " +
mertekegyseg + "\n";
                        }

                    }
                }
                // Kiírja az eredményt
                System.out.println(kiirtSor);

            } catch (ParserConfigurationException | SAXException | IOException e) { e.printStackTrace();
            }
        }
}
```

2.d feladat: Adatírás

Ebben a feladatban létrehozok áruházakat, beszállítókat, áruház-beszállító kapcsolatokat, beszállító raktárakat, áruház raktárakat, valamint akciós termékeket és ezeket struktúráltan iratom ki a konzolra, majd ugyanígy fájlba. A működési elve hasonló az adatolvasás feladatéhoz, mivel egy dokumentumot kapnak a kiíró függvények és azt dolgozzák fel. Azonban itt a dokumentumot nem beolvasom, hanem felépítem saját függvények segítségével. Az *add* kezdetű függvények egészítik ki a dokumentumot a megfelelő elemekkel, teljesen tetszőleges névvel, illetve további tulajdonságokkal lehet ellátni egy-egy elemet (nyílván a struktúra határain belül). A *read* kezdetű függvények pedig paraméterként megkapják a feltöltött dokumentumot és kiírják a konzolra, valamint fájlba. A fájl amibe kiír a következő: *XMLAQYO8L_1.xml*. Fontos még megjegyezni, hogy az akciós termékek árai HashMap-ekben vannak tárolva, ezáltal egyszerű kivenni belőlük a pénznemet és az adott valutában az értékét.

```
package hu.domparse.aqyo8l;

import java.io.File;
import java.io.PrintStream;
import java.io.PrintWriter;
import java.util.HashMap;

import javax.xml.parsers.DocumentBuilder;
```

```
import javax.xml.parsers.DocumentBuilderFactory; import
org.w3c.dom.Document; import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

public class DomWriteAQYO8L {
	public static void main(String args[]) {
		try {
			// DocumentFactory inicializálása
			DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();

			// DocumentBuilder inicializálása
			DocumentBuilder builder = factory.newDocumentBuilder();

			// Létrehozza a dokumentumot
			Document doc = builder.newDocument();

			// Gyökér elem létrehozása
			Element rootElement = doc.createElement("Aruhaz-beszallito_AQYO8L");

			rootElement.setAttribute("xmlns:xsi",
"http://www.w3.org/2001/XMLSchema-instance");
			rootElement.setAttribute("xsi:noNamespaceSchemaLocation",
"XMLSchemaAQYO8L.xsd");

			//	Gyökér elem hozzáadása a dokumentumhoz
			doc.appendChild(rootElement);

			//	Áruházak "doboz" elem létrehozása
			Element aruhazak = createBoxElement(doc, rootElement, "Aruhazak");

			// Beszállítók "doboz" elem létrehozása
			Element beszallitok = createBoxElement(doc, rootElement, "Beszallitok");

			// Raktárak "doboz" elem létrehozása
			Element raktarak = createBoxElement(doc, rootElement, "Raktarak");

			// Akciós termékek "doboz" elem létrehozása
			Element akcios_termekek = createBoxElement(doc, rootElement,
"Akcios_termekek");

			// Áruházak hozzáadása
			addAruhaz(doc, aruhazak, "1", "Coop", "3534, Miskolc, Szentpáli u, 1"); addAruhaz(doc, aruhazak,
			"2", "Tesco", "3535, Miskolc, Példa u, 2"); addAruhaz(doc, aruhazak, "3", "Auchan", "1001,
			Budapest, Mint u, 3"); addAruhaz(doc, aruhazak, "4", "Spar", "2345, Székesfehérvár, Próba u,
4");
```

MISKOLC, 2025

```
                addAruhaz(doc, aruhazak, "5", "Lidl", "9876, Szendrő, Teszt u, 5");

                // Beszállítók hozzáadása
                                addBeszallito(doc, beszallitok, "1", "Kaja futár", "Élelmiszer", "nap",
"2");
                addBeszallito(doc, beszallitok, "2", "Szomjoltó", "Üdítő", "nap", "1"); addBeszallito(doc, beszallitok,
                "3", "Szomjoltó 2", "Üdítő", "nap",
"1");
                addBeszallito(doc, beszallitok, "4", "Adalék-élelmiszer", "Élelmiszer", "nap", "2");
                                addBeszallito(doc, beszallitok, "5", "Csokitanya", "Élelmiszer", "nap",
"3");

                // Raktár termékek hozzáadása
                addAruhazRaktarTermek(doc, raktarak, "1", "1", "Nescafe", "100", "Kávé", "1000",
"HUF");
                addAruhazRaktarTermek(doc, raktarak, "1", "2", "Coca-Cola", "100", "Üdítő", "500",
"HUF");
                addAruhazRaktarTermek(doc, raktarak, "2", "3", "Pepsi", "100", "Üdítő", "500", "HUF");
                addAruhazRaktarTermek(doc, raktarak, "3", "4", "Hellmann's", "100", "Élelmiszer",
"1000", "HUF");
                addAruhazRaktarTermek(doc, raktarak, "3", "5", "Mars", "100", "Élelmiszer",
"500", "HUF");

                addBeszallitoRaktarTermek(doc, raktarak, "1", "1", "Nescafe", "178",
"Kávé");
                addBeszallitoRaktarTermek(doc, raktarak, "2", "2", "Coca-Cola", "21",
"Üdítő");
                addBeszallitoRaktarTermek(doc, raktarak, "3", "3", "Pepsi", "76",
"Üdítő");
                addBeszallitoRaktarTermek(doc, raktarak, "4", "4", "Hellmann's", "134", "Élelmiszer");
                addBeszallitoRaktarTermek(doc, raktarak, "5", "5", "Mars", "200", "Élelmiszer");

                //    Akciós termékek hozzáadása
                //    Akciós termékek árait HashMap-ben tárolom, így találtam a
legegyszerűbbnek a felépítését
                HashMap<String, String> normalPrices1 = new HashMap<String, String>();
                normalPrices1.put("HUF", "1000");
                normalPrices1.put("EUR", "3");
                HashMap<String, String> discountPrices1 = new HashMap<String,
String>();
                discountPrices1.put("HUF",                "500");
                discountPrices1.put("EUR", "1.5");
                addAkciosTermek(doc, akcios_termekek, "1", "1", "Nescafe", "Finom, lágy és krémes: Ez a mi
kávénk! Nescafe",
                        "Kávé", normalPrices1,
```

```
                        discountPrices1);

                HashMap<String, String> normalPrices2 = new HashMap<String, String>();
                normalPrices2.put("HUF", "500");
                normalPrices2.put("EUR", "1.5");
                HashMap<String, String> discountPrices2 = new HashMap<String,
String>();
                discountPrices2.put("HUF", "250");
                discountPrices2.put("EUR", "0.75");
                addAkciosTermek(doc, akcios_termekek, "2", "2", "Coca-Cola",
                        "A Coca-Cola egy szénsavas üdítőital, amelyet a Coca-Cola Company
gyárt.", "Üdítő", normalPrices2,
                        discountPrices2);

                HashMap<String, String> normalPrices3 = new HashMap<String, String>();
                normalPrices3.put("HUF", "500");
                normalPrices3.put("EUR", "1.5");
                HashMap<String, String> discountPrices3 = new HashMap<String,
String>();
                discountPrices3.put("HUF", "250");
                discountPrices3.put("EUR", "0.75");
                addAkciosTermek(doc, akcios_termekek, "3", "3", "Pepsi",
                        "A Pepsi egy szénsavas üdítőital, amelyet a PepsiCo gyárt.", "Üdítő",
normalPrices3,
                        discountPrices3);

                // Áruház-beszállító kapcsolatok
                addAruhazBeszallitoKapcsolat(doc, rootElement, "1", "1", "125");
                addAruhazBeszallitoKapcsolat(doc, rootElement, "1", "2", "200");
                addAruhazBeszallitoKapcsolat(doc, rootElement, "2", "3", "100");
                addAruhazBeszallitoKapcsolat(doc, rootElement, "3", "4", "20");
                addAruhazBeszallitoKapcsolat(doc, rootElement, "3", "5", "300");

                //    Dokumentum kiírása, mentése
                File outputFile = new
File("XMLTaskAQYO8L\\2.feladat\\XMLAQYO8L_1.xml");

                PrintWriter file = new PrintWriter(outputFile, "UTF-8");

                printHeader(doc, file);

                readAruhaz(doc, file);

                readBeszallito(doc, file);

                readAruhazBeszallito(doc, file);

                readRaktarak(doc, file);
```

```
                readAkciosTermekek(doc, file);

                                printToFileAndConsole("</Aruhaz-beszallito_AQYO8L>", System.out, file);

                file.close();

            } catch (Exception e) {
                e.printStackTrace();
            }
        }

        /* Kiíró, adatfeldolgozó rész kezdete */

        // Kiíró metódus
        private static void printToFileAndConsole(final String msg, PrintStream console, PrintWriter
file) {
            console.println(msg);
            file.println(msg);
        }

        // Fejrész elkészítő metódus
        private static void printHeader(Document doc, PrintWriter file) {
            printToFileAndConsole("<?xml version=\"1.0\" encoding=\"UTF-8\"?>",
System.out, file);
            printToFileAndConsole(
                    "<Aruhaz-beszallito_AQYO8L
xmlns:xs=\"http://www.w3.org/2001/XMLSchema-instance\"
xs:noNamespaceSchemaLocation=\"XMLSchemaAQYO8L.xsd\">",
                    System.out, file);
        }

        // Elem kiírás formázó metódus
        private static void printElement(String elementName, String content, PrintWriter file) {
            printToFileAndConsole("                        <" + elementName + ">" + content + "</"
+  elementName + ">", System.out, file);
        }

        // Cím kiírás formázó metódus
        private static void printCim(Element cimElement, PrintWriter file) {
            printToFileAndConsole(" <" + cimElement.getNodeName() + ">", System.out, file);
            printToFileAndConsole("                        <" +
cimElement.getElementsByTagName("Iranyitoszam").item(0).getNodeName()
                    +">"+
cimElement.getElementsByTagName("Iranyitoszam").item(0).getTextContent() + "</"
                    +
cimElement.getElementsByTagName("Iranyitoszam").item(0).getNodeName() + ">",
```

```
                        System.out, file);
            printToFileAndConsole("                                <" +
cimElement.getElementsByTagName("Telepules").item(0).getNodeName()
                        +">"+
cimElement.getElementsByTagName("Telepules").item(0).getTextContent() + "</"
                        +
cimElement.getElementsByTagName("Telepules").item(0).getNodeName() + ">", System.out,
                        file);
            printToFileAndConsole("                                <" +
cimElement.getElementsByTagName("Utca").item(0).getNodeName() + ">"
                        +  cimElement.getElementsByTagName("Utca").item(0).getTextContent()
+  "</"
                        + cimElement.getElementsByTagName("Utca").item(0).getNodeName() +
">",
                        System.out, file);
            printToFileAndConsole("                                <" +
cimElement.getElementsByTagName("Hazszam").item(0).getNodeName()
                        +">"+
cimElement.getElementsByTagName("Hazszam").item(0).getTextContent() + "</"
                        +  cimElement.getElementsByTagName("Hazszam").item(0).getNodeName()
+  ">", System.out, file);
            printToFileAndConsole(" </" + cimElement.getNodeName() + ">", System.out, file);
        }

        // Áruházakat beolvasó metódus
        private static void readAruhaz(Document document, PrintWriter file) { NodeList aruhazList =
            document.getElementsByTagName("Aruhaz"); printToFileAndConsole(" <Aruhazak>",
            System.out, file);
            for (int temp = 0; temp < aruhazList.getLength(); temp++) { Node node =
                aruhazList.item(temp);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element aruhazElement = (Element) node; String nev =
aruhazElement.getElementsByTagName("Nev").item(0).getTextContent(); String aruhazId =
                    aruhazElement.getAttribute("aruhazid"); Element cimElement = (Element)
aruhazElement.getElementsByTagName("Cim").item(0);
```

```
                    printToFileAndConsole(" <Aruhaz aruhazid=\"" + aruhazId + "\">", System.out, file);
                    printElement("Nev", nev, file);
                    printCim(cimElement, file);
                    printToFileAndConsole("                        </Aruhaz>", System.out, file);
                }
            }
            printToFileAndConsole("            </Aruhazak>", System.out, file);
        }
```

```
    // Beszállítókat beolvasó metódus
    private static void readBeszallito(Document document, PrintWriter file) {
        NodeList beszallitoList = document.getElementsByTagName("Beszallito");
        printToFileAndConsole(" <Beszallitok>", System.out, file);
        for (int temp = 0; temp < beszallitoList.getLength(); temp++) {
            Node node = beszallitoList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element beszallitoElement = (Element) node; String nev =
beszallitoElement.getElementsByTagName("Nev").item(0).getTextContent();
                String beszallitoid =
beszallitoElement.getAttribute("beszallitoid");
                String termekKategoria =
        beszallitoElement.getElementsByTagName("Termek_kategoria").item(0)
                                .getTextContent();
                Element atlagosSzallitasiIdoElement = (Element) beszallitoElement
                        .getElementsByTagName("Atlagos_szallitasi_ido").item(0);
                String atlagosSzallitasiIdoString =
atlagosSzallitasiIdoElement.getTextContent();
                String atlagosSzallitasiIdoMertekegyseg =
atlagosSzallitasiIdoElement.getAttribute("mertekegyseg");

                printToFileAndConsole("                    <Beszallito beszallitoid=\"" +
beszallitoid + "\">", System.out, file);
                printElement("Nev", nev, file);
                printElement("Termek_kategoria", termekKategoria, file);
                printToFileAndConsole(
                        "                                <Atlagos_szallitasi_ido mertekegyseg=\"" +
atlagosSzallitasiIdoMertekegyseg
                                                + "\">" + atlagosSzallitasiIdoString +
"</Atlagos_szallitasi_ido>",
                        System.out, file);
                printToFileAndConsole("                    </Beszallito>", System.out, file);
            }
        }
        printToFileAndConsole("                                </Beszallitok>", System.out, file);
    }

    // Áruház-Beszállító kapcsolatokat beolvasó metódus
    private static void readAruhazBeszallito(Document document, PrintWriter file) { NodeList
        aruhazBeszallitoList = document.getElementsByTagName("Aruhaz-
Beszallito");
        for (int temp = 0; temp < aruhazBeszallitoList.getLength(); temp++) { Node node =
            aruhazBeszallitoList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) { Element
                aruhazBeszallitoElement = (Element) node;
                String atlagos_Rendelt_Arumennyiseg = aruhazBeszallitoElement
                        .getElementsByTagName("Atlagos_Rendelt_Arumennyiseg").item(
0).getTextContent();
```

```
                String beszallitoid =
aruhazBeszallitoElement.getAttribute("beszallitoid");
                                String aruhazid = aruhazBeszallitoElement.getAttribute("aruhazid");

                printToFileAndConsole(" <Aruhaz-Beszallito aruhazid=\"" + aruhazid + "\"
beszallitoid=\""
                        +  beszallitoid + "\">", System.out, file);
                printElement("Atlagos_Rendelt_Arumennyiseg",
atlagos_Rendelt_Arumennyiseg, file);
                printToFileAndConsole("                </Aruhaz-Beszallito>", System.out,
file);
            }
        }
    }

    // Raktárakat beolvasó metódus
    private static void readRaktarak(Document document, PrintWriter file) { NodeList
        aruhazRaktarTermekList =
document.getElementsByTagName("Aruhaz_Raktar_Termek"); NodeList
        beszallitoRaktarTermekList =
document.getElementsByTagName("Beszallito_Raktar_Termek");
        printToFileAndConsole(" <Raktarak>", System.out, file);

        for (int temp = 0; temp < aruhazRaktarTermekList.getLength(); temp++) { Node node =
            aruhazRaktarTermekList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) { Element
                aruhaRaktarElement = (Element) node;
                String termekid = aruhaRaktarElement.getAttribute("termekid");
                String aruhazid = aruhaRaktarElement.getAttribute("aruhazid");
                String nev =
aruhaRaktarElement.getElementsByTagName("Nev").item(0).getTextContent(); String darabszam
                =
aruhaRaktarElement.getElementsByTagName("Darabszam").item(0).getTextContent(); String kategoria =
aruhaRaktarElement.getElementsByTagName("Kategoria").item(0).getTextContent(); Element arElement =
                (Element)
aruhaRaktarElement.getElementsByTagName("Ar").item(0); String ar =
                arElement.getTextContent();
                        String penznem = arElement.getAttribute("penznem");

                printToFileAndConsole(" <Aruhaz_Raktar_Termek aruhazid=\"" + aruhazid + "\"
termekid=\""
                        +  termekid + "\">", System.out, file);
                printElement("Nev", nev, file); printElement("Darabszam",
                darabszam, file); printElement("Kategoria", kategoria, file);
                printToFileAndConsole("                        <Ar penznem=\"" + penznem +
"\">" + ar + "</Ar>", System.out, file);
```

MISKOLC, 2025

```
                    printToFileAndConsole("                    </Aruhaz_Raktar_Termek>", System.out,
file);
                }
            }

            for (int temp2 = 0; temp2 < beszallitoRaktarTermekList.getLength(); temp2++) {
                Node node2 = beszallitoRaktarTermekList.item(temp2); if
                (node2.getNodeType() == Node.ELEMENT_NODE) {
                    Element beszallitoRaktarElement = (Element) node2;
                    String termekid = beszallitoRaktarElement.getAttribute("termekid"); String beszallitoid =
beszallitoRaktarElement.getAttribute("beszallitoid"); String nev =
beszallitoRaktarElement.getElementsByTagName("Nev").item(0).getTextContent(); String darabszam =
            beszallitoRaktarElement.getElementsByTagName("Darabszam").item(0)
                                    .getTextContent();
                    String kategoria =
beszallitoRaktarElement.getElementsByTagName("Kategoria").item(0)
                                .getTextContent();

                    printToFileAndConsole(
                                "                <Beszallito_Raktar_Termek beszallitoid=\"" +
beszallitoid + "\" termekid=\""
                                        +  termekid + "\">",
                                System.out, file);
                    printElement("Nev", nev, file);
                    printElement("Darabszam", darabszam, file);
                    printElement("Kategoria", kategoria, file);
                    printToFileAndConsole(" </Beszallito_Raktar_Termek>", System.out, file);
                }
            }
            printToFileAndConsole("            </Raktarak>", System.out, file);
        }

        // Akciós termékeket beolvasó metódus
        private static void readAkciosTermekek(Document document, PrintWriter file) { NodeList
            akciosTermekekList =
document.getElementsByTagName("Akcios_Termek");
            printToFileAndConsole("            <Akcios_Termekek>", System.out, file);
            for (int temp = 0; temp < akciosTermekekList.getLength(); temp++) { Node node =
                akciosTermekekList.item(temp);
                if (node.getNodeType() == Node.ELEMENT_NODE) { Element
                    akciosTermekElement = (Element) node;
                    String termekid = akciosTermekElement.getAttribute("termekid"); String
                    akciostermekid =
akciosTermekElement.getAttribute("akciostermekid");
```

```
                    String nev =
akciosTermekElement.getElementsByTagName("Nev").item(0).getTextContent();
                    String leiras =
akciosTermekElement.getElementsByTagName("Leiras").item(0).getTextContent();
                    String kategoria =
akciosTermekElement.getElementsByTagName("Kategoria").item(0).getTextContent();

                        NodeList eredetiArList =
        akciosTermekElement.getElementsByTagName("Eredeti_ar");
                        NodeList akciosArList =
       akciosTermekElement.getElementsByTagName("Akcios_ar");
```

```
                    printToFileAndConsole("                    <Akcios_Termek termekid=\"" +
termekid + "\" akciostermekid=\""
                            +  akciostermekid + "\">", System.out, file);
                    printElement("Nev", nev, file); printElement("Leiras", leiras, file);
                    printElement("Kategoria", kategoria, file);

                    printToFileAndConsole("                        <Arak>", System.out, file);

                    for (int temp2 = 0; temp2 < eredetiArList.getLength(); temp2++) { Node node2 =
                        eredetiArList.item(temp2);
                        if (node2.getNodeType() == Node.ELEMENT_NODE) {
                            Element eredetiAr = (Element) node2;
                            String penznem = eredetiAr.getAttribute("penznem"); String ar =
                            eredetiAr.getTextContent(); printToFileAndConsole(
                                    "                            <Eredeti_ar penznem=\"" + penznem
+ "\">" + ar + "</Eredeti_ar>",
                                            System.out, file);
                        }
                    }

                    for (int temp3 = 0; temp3 < akciosArList.getLength(); temp3++) { Node node3 =
                        akciosArList.item(temp3);
                        if (node3.getNodeType() == Node.ELEMENT_NODE) {
                            Element akciosAr = (Element) node3;
                            String penznem = akciosAr.getAttribute("penznem"); String ar =
                            akciosAr.getTextContent(); printToFileAndConsole(
                                    "                            <Akcios_ar penznem=\"" + penznem +
"\">" + ar + "</Akcios_ar>",
                                            System.out, file);
                        }
                    }

                    printToFileAndConsole("                        </Arak>", System.out, file);
```

```
                        printToFileAndConsole("                        </Akcios_Termek>", System.out, file);
                    }
                }
                printToFileAndConsole("            </Akcios_Termekek>", System.out, file);
        }

        /* Hozzáadó rész kezdete */

        // "Doboz" elem létrehozása
        private static Element createBoxElement(Document doc, Element rootElement, String name) {
                Element box = doc.createElement(name);
                rootElement.appendChild(box);

                return box;
        }

        // Áruház hozzáadása
        private static void addAruhaz(Document doc, Element rootElement, String aruhaz_id, String
name, String address) {
                Element aruhaz = doc.createElement("Aruhaz");
                aruhaz.setAttribute("aruhazid", aruhaz_id);

                Element nevElement = createElementAndAddToDoc(doc, "Nev", name); Element
                cimElement = doc.createElement("Cim");
                Element iranyitoszamElement = createElementAndAddToDoc(doc, "Iranyitoszam",
address.split(",")[0].trim());
                Element telepulesElement = createElementAndAddToDoc(doc, "Telepules",
address.split(",")[1].trim());
                Element utcaElement = createElementAndAddToDoc(doc, "Utca",
address.split(",")[2].trim());
                Element hazszamElement = createElementAndAddToDoc(doc, "Hazszam",
address.split(",")[3].trim());
                aruhaz.appendChild(nevElement);
                aruhaz.appendChild(cimElement);
                cimElement.appendChild(iranyitoszamElement);
                cimElement.appendChild(telepulesElement);
                cimElement.appendChild(utcaElement);
                cimElement.appendChild(hazszamElement);

                rootElement.appendChild(aruhaz);
        }

        // Beszállító hozzáadása
        private static void addBeszallito(Document doc, Element rootElement, String beszallito_id, String
name,
                        String category, String measureString, String time) { Element beszallito
                = doc.createElement("Beszallito"); beszallito.setAttribute("beszallitoid",
                beszallito_id);
```

```
            Element nevElement = createElementAndAddToDoc(doc, "Nev", name); Element
            kategoriaElement = createElementAndAddToDoc(doc,
"Termek_kategoria", category);
            Element szallitasiIdoElement = createElementAndAddToDoc(doc,
"Atlagos_szallitasi_ido", time);
            szallitasiIdoElement.setAttribute("mertekegyseg", measureString);

            beszallito.appendChild(nevElement);
            beszallito.appendChild(kategoriaElement);
            beszallito.appendChild(szallitasiIdoElement);

            rootElement.appendChild(beszallito);
        }

        // Áruház-Berszállító kapcsolat hozzáadása
        private static void addAruhazBeszallitoKapcsolat(Document doc, Element rootElement, String
beszallito_id,
                String aruhaz_id, String orderedGoods) {
            Element aruhazBeszallitoKapcs = doc.createElement("Aruhaz-Beszallito");
            aruhazBeszallitoKapcs.setAttribute("aruhazid", aruhaz_id);
            aruhazBeszallitoKapcs.setAttribute("beszallitoid", beszallito_id);

            Element rendeltTermekElement = createElementAndAddToDoc(doc,
"Atlagos_Rendelt_Arumennyiseg", orderedGoods);
            aruhazBeszallitoKapcs.appendChild(rendeltTermekElement);

            rootElement.appendChild(aruhazBeszallitoKapcs);
        }

        // Áruház raktár termék hozzáadása
        private static void addAruhazRaktarTermek(Document doc, Element rootElement, String aruhaz_id,
String termek_id,
                String name, String quantity, String category, String price, String priceExchange) {
            Element aruhazRaktarTermek = doc.createElement("Aruhaz_Raktar_Termek");
            aruhazRaktarTermek.setAttribute("aruhazid", aruhaz_id);
            aruhazRaktarTermek.setAttribute("termekid", termek_id);

            Element nevElement = createElementAndAddToDoc(doc, "Nev", name); Element
            quantityElement = createElementAndAddToDoc(doc, "Darabszam",
quantity);
            Element kategoriaElement = createElementAndAddToDoc(doc, "Kategoria", category);
            Element arElement = createElementAndAddToDoc(doc, "Ar", price);
            arElement.setAttribute("penznem", priceExchange);

            aruhazRaktarTermek.appendChild(nevElement);
            aruhazRaktarTermek.appendChild(quantityElement);
```

```
            aruhazRaktarTermek.appendChild(kategoriaElement);
            aruhazRaktarTermek.appendChild(arElement);

            rootElement.appendChild(aruhazRaktarTermek);
        }

        // Beszállító raktár termék hozzáadása
        private static void addBeszallitoRaktarTermek(Document doc, Element rootElement, String
beszallito_id,
                    String termek_id, String name, String quantity, String category) { Element
            beszallitoRaktarTermek =
doc.createElement("Beszallito_Raktar_Termek");
            beszallitoRaktarTermek.setAttribute("beszallitoid", beszallito_id);
            beszallitoRaktarTermek.setAttribute("termekid", termek_id);

            Element nevElement = createElementAndAddToDoc(doc, "Nev", name); Element
            quantityElement = createElementAndAddToDoc(doc, "Darabszam",
quantity);
            Element kategoriaElement = createElementAndAddToDoc(doc, "Kategoria", category);

            beszallitoRaktarTermek.appendChild(nevElement);
            beszallitoRaktarTermek.appendChild(quantityElement);
            beszallitoRaktarTermek.appendChild(kategoriaElement);

            rootElement.appendChild(beszallitoRaktarTermek);
        }

        // Akciós termék hozzáadása
        private static void addAkciosTermek(Document doc, Element rootElement, String akcios_termek_id,
String termek_id,
                    String name, String description, String category, HashMap<String, String>
normalPrices,
                    HashMap<String, String> discountPrices) {
            Element akciosTermek = doc.createElement("Akcios_Termek");
            akciosTermek.setAttribute("akciostermekid", akcios_termek_id);
            akciosTermek.setAttribute("termekid", termek_id);

            Element nevElement = createElementAndAddToDoc(doc, "Nev", name); Element
            descriptionElement = createElementAndAddToDoc(doc, "Leiras",
description);
            Element kategoriaElement = createElementAndAddToDoc(doc, "Kategoria", category);

            Element arakElement = doc.createElement("Arak");

            for (String key : normalPrices.keySet()) {
                Element arElement = createElementAndAddToDoc(doc, "Eredeti_ar",
normalPrices.get(key));
```

```
                arElement.setAttribute("penznem", key);
                arakElement.appendChild(arElement);
            }

            for (String key : discountPrices.keySet()) {
                Element arElement = createElementAndAddToDoc(doc, "Akcios_ar",
discountPrices.get(key));
                arElement.setAttribute("penznem", key);
                arakElement.appendChild(arElement);
            }

            akciosTermek.appendChild(nevElement);
            akciosTermek.appendChild(descriptionElement);
            akciosTermek.appendChild(kategoriaElement);
            akciosTermek.appendChild(arakElement);

            rootElement.appendChild(akciosTermek);
        }

        // Elem létrehozása és dokumentumhoz adása
        private static Element createElementAndAddToDoc(Document doc, String name, String value) {
            Element element = doc.createElement(name);
            element.appendChild(doc.createTextNode(value)); return element;
        }

}
```

3. feladat:

3.a feladat: MongoDB adatbázis

Adattároláshoz egy nem relációs adatbázist, a MongoDB-t választottam. Az adatbázis egyedisége, hogy dokumentumokként menti el az objektumot. Egy-egy ilyen dokumentum JSON formátumot vesz fel, ezáltal egyszerű kiolvashatóságot és bővíthetőséget tesz lehetővé.
A dokumentumokat kollekciókban tárolja, amik lényegében egy keretbe foglalják a dokumentumokat. Egy-egy kollekcióra megkötéseket lehet létrehozni, amiket be lehet állítani hogy cask az új dokumentumok esetében nézze, vagy futtassa le a meglévő dokumentumokra is.

A feladathoz két kollekció készült: users és posts.
A users kollekció tárolja a felhasználókat. Egy user dokumentum két mezővel rendelkezik: username és password. Mindkét mező megadása kötelező, a validator üres dokumentumot nem enged beszúrni.
A felhasználó validátora:

```
{
  $jsonSchema: {
    bsonType: 'object',
    required: [
      'username',
      'password'
    ],
    properties: {
      username: {
        bsonType: 'string',
        description: 'A felhasználónévnek szövegesnek kell lennie és kötelező megadni!'
      },
      password: {
        bsonType: 'string',
```

```
      description: 'A jelszó megadása kötelező.'
    }
   }
  }
}
```

A postos kollekcióban kerülnek tárolásra a felhasználók által létrehozott blog bejegyzések. Két adattagal rendelkezik egy-egy dokumentum: cím és tartalom. A kollekció validátora úgy van beállítva, hogy mindkettő megadása kötelező.
A posztok validaátora:

```
{
 $jsonSchema: {
  bsonType: 'object',
  required: [
   'title',
   'content'
  ],
  properties: {
   title: {
    bsonType: 'string',
    description: 'Cim megadása kötelező!'
   },
   content: {
    bsonType: 'string',
    description: 'Tartalom megadása kötelező!'
   }
  }
 }
}
```

3.b feladat: Python Flask backend

A webalkalmazás backendjét a Flask keretrendszerben valósítottam meg, ami Python nyelven íródott. A végpontok a következőek:

- bejelentkező végpont
- regisztrációs végpont
- poszt felvitel végpont
- poszt törlés végpont
- posztok lekérdezése végpont
- más ember posztjainak lekérdezése végpont
- poszt módositás végpont
- token ellenőrző végpont

Bejelentkezéskor a backend vissza ad egy JWT tokent, ami egy óra időtartamig él, ezt követően inaktívvá válik és újat kell kérni (újra be kell bejelntkezni).
A backend a bejelentkező és regisztrációs végpontokon kívül minden HTTP kérésnél elvárja a token meglétét és validságát.
A token ellenőrző végponton kívül az összes végpont végez adatbázis műveletet valamilyen formában.

3.c feladat: HTML-JS-CSS frontend

A webalkalmazás frontendje sima HTML-JS-CSS-ben lett megvalósítva, kihasználva a JavaScript aszinkron előnyeit. Minden HTTP kérés async kérésként fut le, tehát bevárásra kerül a kérés eredménye.

A stack választásának legfőbb indoka az egyszerűség volt, nem gondoltam szükségesnek egy keretrendszer bevonását.