

# JEGYZŐKÖNYV

Webes adatkezelő környezetek

Féléves feladat

Pszichiátria Nyilvántartó Rendszer

Készítette: Berki Viktor

Neptunkód: ZY5P7F

Dátum: 2025. december 1.

Miskolc, 2025

## **Tartalomjegyzék**

<b>1. XML alapú adatkezelés tervezése és megvalósítása</b>	<b>2</b>
1.1. Az adatbázis ER modell tervezése . . . . .	2
1.2. Az adatbázis konvertálása XDM modellre . . . . .	3
1.3. Az XDM modell alapján XML dokumentum készítése . . . . .	3
1.4. Az XML dokumentum alapján XMLSchema készítése . . . . .	4
<b>2. DOM program készítése</b>	<b>5</b>
2.1. Adatolvasás . . . . .	5
2.2. Adat-lekérdezés . . . . .	5
2.3. Adatmódosítás . . . . .	6

# Bevezetés

A féléves feladat során egy pszichiátriai osztály nyilvántartó rendszerét valósítottam meg. A rendszer célja az osztályok, orvosok, páciensek, a hozzájuk tartozó kórlapok és a felírt gyógyszerek adatainak strukturált tárolása és kezelése XML alapokon. [cite: 10]

## 1 XML alapú adatkezelés tervezése és megvalósítása

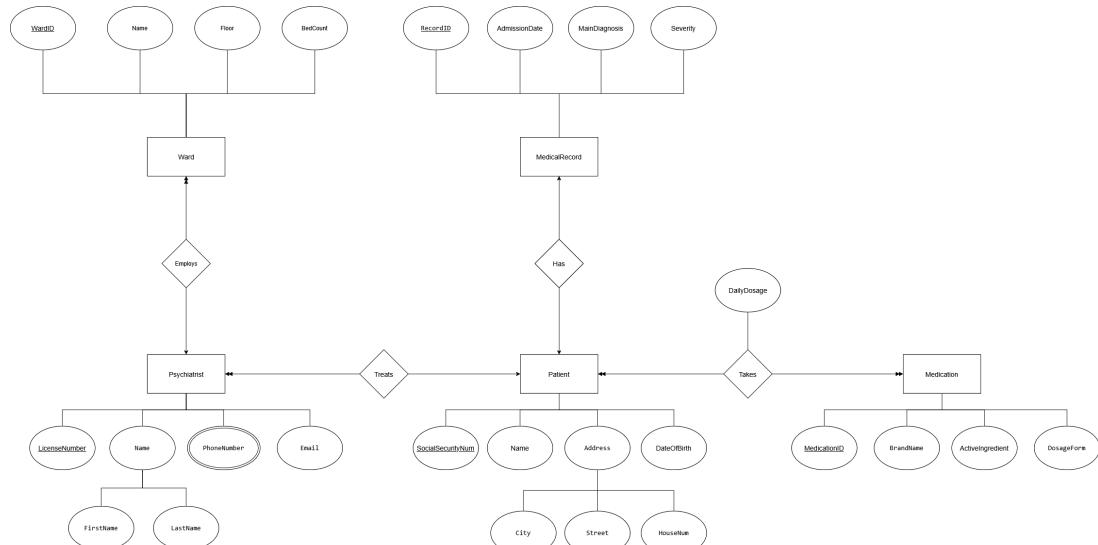
### 1.1 Az adatbázis ER modell tervezése

A tervezés során öt egyedet definiáltam: **Ward** (Osztály), **Psychiatrist** (Pszichiáter), **Patient** (Páciens), **MedicalRecord** (Kórlap) és **Medication** (Gyógyszer).

A kapcsolatok a következők:

- **1:N kapcsolat:** Egy osztályon több orvos dolgozik, egy orvos több beteget kezel.
- **1:1 kapcsolat:** Egy betegnek pontosan egy kórlapja van.
- **M:N kapcsolat:** A betegek és gyógyszerek között (a beteg szedi a gyógyszert), melyet a *Takes* kapcsolat valósít meg *DailyDosage* tulajdonsággal.

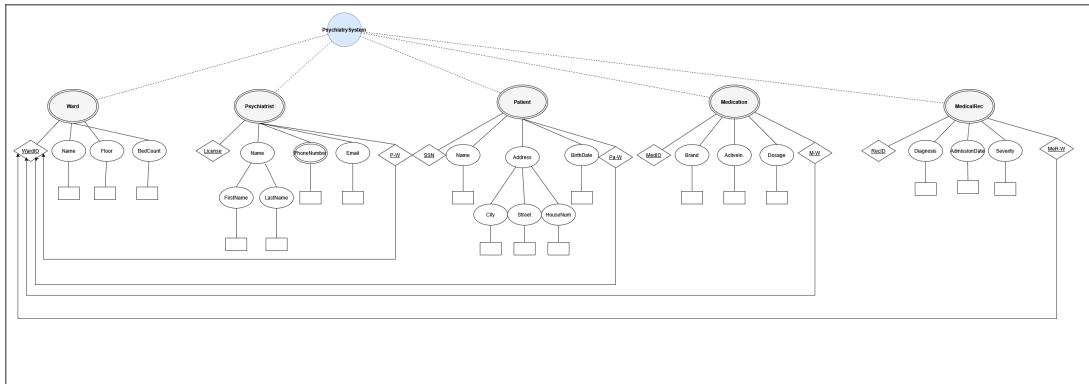
Az egyedek rendelkeznek kulcsokkal (PK), összetett (pl. Lakcím, Név) és többértékű (pl. Telefonszám) tulajdonságokkal is. [cite: 15]



1. ábra. A rendszer ER modellje

## 1.2 Az adatbázis konvertálása XDM modellre

Az ER modellt hierarchikus XDM modellé alakítottam át. A "vonalak ne keresztezzék egymást" szabály betartása érdekében a **Medications** listát külön ágon, törzsadatként kezelem. A hierarchia jobb oldala a **Ward → Psychiatrist → Patient** láncolatot követi. A **MedicalRecord** a páciensbe ágyazódik be. Az M:N kapcsolatot a **Prescription** elem oldja meg, amely IDREF attribútummal mutat a bal oldali gyógyszer listára. [cite: 17, 19]



2. ábra. A rendszer XDM modellje

## 1.3 Az XDM modell alapján XML dokumentum készítése

Az XML dokumentum a ‘PsychiatrySystem’-gyökérelemből alatt tárolja az adatokat. minden többszörösen előforduló elemből (pl. Ward, Psychiatrist, Medication) legalább két példány készült. A dokumentum tartalmazza a szükséges attribútumokat (id, ref) és a kommenteket. [cite: 21, 22]

**Fájlnév:** ZY5P7F\_XML.xml

1. Listing. Részlet az XML fájlból

```
<PsychiatrySystem xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="ZY5P7F_XMLSchema.xsd">
    <Medications>
        <Medication id="MED001">
            <BrandName>Xanax SR</BrandName>
            <ActiveIngredient>Alprazolam</ActiveIngredient>
            <DosageForm>Tablet</DosageForm>
        </Medication>
    </Medications>

    <Wards>
        <Ward id="W01">
            <Name>Acute Care Unit</Name>
            <Floor>2</Floor>
            <BedCount>30</BedCount>
            <Psychiatrists>
```

```

<Psychiatrist license="LIC-12345">
    <Name>
        <FirstName>Gregory</FirstName>
        <LastName>House</LastName>
    </Name>
    <Patients>
        <Patient ssn="111-22-3333">
            <Prescription ref="MED002">
                <DailyDosage>1x20mg Morning</DailyDosage>
            </Prescription>
        </Patient>
    </Patients>
</Psychiatrist>
</Psychiatrists>
</Ward>
</Wards>
</PsychiatrySystem>

```

## 1.4 Az XML dokumentum alapján XMLSchema készítése

Az XSD sémában saját egyszerű és összetett típusokat definiáltam (pl. SSNTYPE, AddressType). A kapcsolati integritást xs:key és xs:keyref elemekkel biztosítottam, így a receptek csak létező gyógyszer ID-ra hivatkozhatnak. [cite: 24, 25, 26]

**Fájlnév:** ZY5P7F\_XMLSchema.xsd

2. Listing. Részlet az XSD sémából

```

<xs:complexType name="PrescriptionType">
    <xs:sequence>
        <xs:element name="DailyDosage" type="xs:string"/>
    </xs:sequence>
    <xs:attribute name="ref" type="xs:IDREF" use="required"/>
</xs:complexType>

<xs:key name="MedicationKey">
    <xs:selector xpath="Medications/Medication"/>
    <xs:field xpath="@id"/>
</xs:key>

<xs:keyref name="PrescriptionToMedicationRef" refer="MedicationKey">
    <xs:selector xpath=".//Prescription"/>
    <xs:field xpath="@ref"/>
</xs:keyref>

```

## 2 DOM program készítése

A feladat második részében egy Java alkalmazást készítettem, amely DOM (Document Object Model) segítségével dolgozza fel az XML állományt.

**Projekt adatok:** [cite: 30, 31, 32]

- **Project name:** ZY5P7FDOMParse
- **Package:** zy5p7f.domparse.hu
- **Class names:** ZY5P7FDomRead, ZY5P7FDomQuery, ZY5P7FDomModify
- **XML name:** ZY5P7F\_XML.xml

### 2.1 Adatolvasás

A ZY5P7FDomRead osztály beolvassa a teljes XML dokumentumot a memóriába, majd egy rekurzív metódus segítségével végigjárja a DOM fát. A program blokkos formában írja ki a konzolra az elemeket, attribútumokat és a szöveges tartalmakat. [cite: 33, 37]

3. Listing. Részlet az Adatolvasás kódjából

```
// Rekurzív metódus a fa bejárására
private static void printNode(Node node, String indent) {
    if (node.getNodeType() == Node.ELEMENT_NODE) {
        System.out.print(indent + "Elem: " + node.getNodeName());

        // Attribútumok kiírása
        if (node.hasAttributes()) {
            NamedNodeMap nodeMap = node.getAttributes();
            for (int i = 0; i < nodeMap.getLength(); i++) {
                Node tempNode = nodeMap.item(i);
                System.out.print(" | " + tempNode.getNodeName() + "=" +
                    tempNode.getNodeValue());
            }
        }
        // ... (további logika)
    }
}
```

### 2.2 Adat-lekérdezés

A ZY5P7FDomQuery osztály célzottan kérdez le adatokat az XML-ből. A program kilistázza az összes pácienset (Patient) és a hozzájuk tartozó fő diagnózist (MainDiagnosis). A megvalósítás során NodeList-et és getElementsByTagName metódust használtam. [cite: 39, 43]

#### 4. Listing. Részlet az Adat-lekérdezés kódból

```
NodeList patientList = doc.getElementsByTagName("Patient");
for (int i = 0; i < patientList.getLength(); i++) {
    Node node = patientList.item(i);
    if (node.getNodeType() == Node.ELEMENT_NODE) {
        Element patient = (Element) node;
        String name = patient.getElementsByTagName("Name").item(0).
            getTextContent();
        String ssn = patient.getAttribute("ssn");
        // ... diagnózis lekérése ...
        System.out.println("Páciens: " + name + " (SSN: " + ssn + ")");
    }
}
```

### 2.3 Adatmódosítás

A ZY5P7FDomModify osztály módosítja az XML tartalmát. A program megkeresi a MED001 azonosítójú gyógyszert, és a márkanevét (BrandName) átírja "Xanax SR"-re. A módosítás után az új állapotot konzolra írja, és elmenti egy ZY5P7F\_XML\_Modified.xml nevű fájlba. [cite: 45, 48]

#### 5. Listing. Részlet az Adatmódosítás kódból

```
if ("MED001".equals(medElement.getAttribute("id"))) {
    Node brandNameNode = medElement.getElementsByTagName("BrandName").
        item(0);
    System.out.println("Régi név: " + brandNameNode.getTextContent());

    brandNameNode.setTextContent("Xanax SR"); // Módosítás
    System.out.println("Új név beállítva: Xanax SR");
}
// ... Mentés Transformerrel ...
```