**Korházi Felület Vizsgaremek**

**Miket tartalmaz eddig a vizsgaremek:**

1. **Bejelentkezési Felület**:
   * A felhasználók (orvosok, adminisztrátorok) egy regisztrált felhasználónévvel és jelszóval tudnak belépni a rendszerbe.
   * A bejelentkezési folyamat a biztonságot szem előtt tartva készült, és a jövőben integrálni szeretnénk **kétfaktoros hitelesítést** is.
2. **Weboldal Kódja és Backend Program**:
   * A kórházi rendszer front-end része egy dinamikus és reszponzív weboldal, amely biztosítja a könnyű hozzáférést bármilyen eszközről (számítógép, tablet, okostelefon).
   * A backend egy stabil, megbízható kódalapra épül, amely biztosítja az adatok gyors feldolgozását és tárolását. Jelenleg támogatja:
     + Felhasználói adatok kezelését (orvosok, adminisztrátorok adatbázisa).
     + Alapvető jogosultsági szintek kezelését.

**3. Folyamatban Lévő Fejlesztések**

A projekt fejlesztése még nem teljes, az alábbi funkciók megvalósításán dolgozunk jelenleg:

1. **Páciens-adatkezelés**:
   * Betegek adatainak nyilvántartása, kórtörténet rögzítése.
   * Időpontfoglalási rendszer, amely lehetővé teszi a betegek számára, hogy időpontot foglaljanak orvosokhoz.
2. **Képfeltöltési és Dokumentumkezelési Rendszer**:
   * Laboreredmények, röntgenfelvételek és egyéb dokumentumok feltöltése és megtekintése.
3. **Kétfaktoros Hitelesítés**:
   * A rendszer biztonságának növelése érdekében dolgozunk egy SMS-alapú vagy alkalmazás-alapú kétfaktoros hitelesítési megoldáson.
4. **Egyedi Statisztikák**:
   * Részletes jelentések és statisztikák készítése, amelyek az orvosok és az adminisztrátorok számára könnyebbé teszik a munkát.

**4. Technológiai Hátterünk**

* **Frontend**: HTML5, CSS3, JavaScript (React.js).
* **Backend**: Python (Django vagy Flask keretrendszer), adatbázis-kezelés MySQL vagy PostgreSQL segítségével.
* **Biztonság**: SSL/TLS titkosítás, jelszavak biztonságos kezelése (bcrypt, Argon2).

A program jelenleg **fejlesztési alatt áll**, így néhány funkció még nem érhető el, és hibák előfordulhatnak.

**Mik is azok a programok amiket használunk a vizsgaremekben:**

**-html**

**-javascript**

**-react**

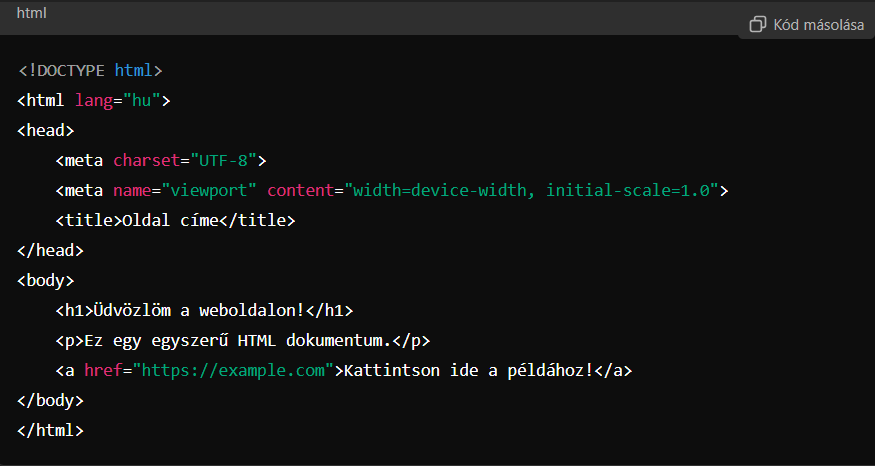
**-mysql**

**HTML:**

A **HTML** egy jelölőnyelv, amelyet weboldalak készítésére használnak. Ez a nyelv az internetes tartalom strukturálására és megjelenítésére szolgál, például szövegek, képek, linkek, űrlapok és egyéb elemek formázására.

A HTML dokumentumok egyszerű szöveges fájlok, amelyek speciális **tag-ek** segítségével jelölik meg a különböző tartalmi elemeket. Minden HTML fájl a **<html>** taggel kezdődik, és a **</html>** taggel zárul. A dokumentumon belül további fontos elemek találhatók, mint például:

* **<head>**: Az oldal metaadatai (pl. cím, karakterkódolás) találhatók itt.
* **<body>**: Az oldal fő tartalma itt helyezkedik el.
* **<h1>, <h2>, <h3>, stb.**: Címsorok különböző szintjei.
* **<p>**: Bekezdés.
* **<a>**: Hiperlinkek létrehozása.
* **<img>**: Képek beszúrása.
* **<ul>, <ol>, <li>**: Listaelemek létrehozása.

itt egy html alap példa: 

így kezdődik egy alap html weboldal.

**Javascript:**

A **JavaScript** egy magas szintű, dinamikus programozási nyelv, amelyet elsősorban weboldalak interaktivitásának és dinamikus viselkedésének biztosítására használnak. A JavaScript a **kliense oldali** (böngészőben futó) programozás egyik alapvető eszköze, de a **szerveroldalon** is alkalmazható (például Node.js segítségével).

### Főbb jellemzők:

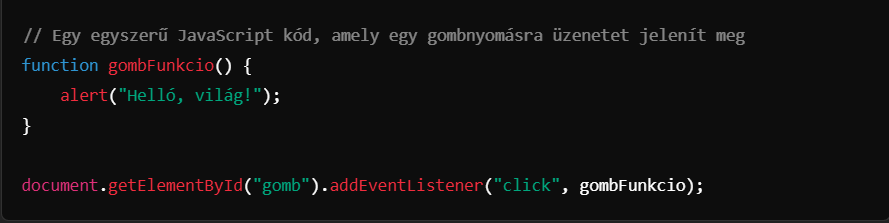
* **Interaktivitás**: Lehetővé teszi a felhasználói interakciók kezelését, például űrlapok validálása, gombnyomások kezelése, dinamikus tartalom frissítése anélkül, hogy újra kellene tölteni az oldalt.
* **Dinamikus DOM manipuláció**: A JavaScript képes a weboldal tartalmát (a DOM-ot – Document Object Model) futásidőben módosítani, új elemeket hozzáadni vagy eltávolítani.
* **Aszinkron működés**: A JavaScript képes aszinkron műveletek végrehajtására, például adatbetöltés AJAX segítségével, anélkül, hogy az egész oldal újratöltődne.
* **Széleskörű használat**: Használják kliensoldali programozásra a weboldalakon, illetve szerveroldali alkalmazásokban (Node.js), mobilalkalmazásokban (React Native), játékfejlesztésben, és sok más területen.

A **JavaScript** egy magas szintű, dinamikus programozási nyelv, amelyet elsősorban weboldalak interaktivitásának és dinamikus viselkedésének biztosítására használnak. A JavaScript a **kliense oldali** (böngészőben futó) programozás egyik alapvető eszköze, de a **szerveroldalon** is alkalmazható (például Node.js segítségével).

### Főbb jellemzők:

* **Interaktivitás**: Lehetővé teszi a felhasználói interakciók kezelését, például űrlapok validálása, gombnyomások kezelése, dinamikus tartalom frissítése anélkül, hogy újra kellene tölteni az oldalt.
* **Dinamikus DOM manipuláció**: A JavaScript képes a weboldal tartalmát (a DOM-ot – Document Object Model) futásidőben módosítani, új elemeket hozzáadni vagy eltávolítani.
* **Aszinkron működés**: A JavaScript képes aszinkron műveletek végrehajtására, például adatbetöltés AJAX segítségével, anélkül, hogy az egész oldal újratöltődne.
* **Széleskörű használat**: Használják kliensoldali programozásra a weboldalakon, illetve szerveroldali alkalmazásokban (Node.js), mobilalkalmazásokban (React Native), játékfejlesztésben, és sok más területen.
* **Változók**: Az adatok tárolására szolgáló objektumok, például let, const, vagy var.
* **Funkciók**: Kódblokkok, amelyek újrahasznosíthatók és meghatározzák, hogy mi történjen egy művelet során.
* **Események**: Felhasználói interakciók (pl. kattintás, billentyűleütés) kezelésére szolgáló mechanizmusok.
* **Ciklusok és feltételes utasítások**: Lehetővé teszik az ismétlődő műveletek végrehajtását (pl. for ciklus), és a különböző feltételek kezelését (pl. if).
* **Objektumok és tömbök**: Az adatok strukturálásának eszközei.
* **Aszinkron programozás**: A JavaScript támogatja az aszinkron hívásokat, például a setTimeout, setInterval, illetve Promise vagy async/await segítségével.

itt egy javascript alap:



### Hogyan működik?

A JavaScript kód általában a HTML dokumentumban van elhelyezve, vagy külön fájlokban, és a böngésző végrehajtja azt, amikor az oldal betöltődik. A JavaScript lehetővé teszi, hogy a weboldal interaktívvá váljon, például animációk, dinamikus tartalom, vagy válaszok megjelenítése a felhasználói interakciók alapján.

**react:**

### Főbb jellemzők:

* **Komponens-alapú fejlesztés**: A React a felhasználói felületeket kisebb, izolált, újrahasználható komponensekre bontja. Minden komponens egy önálló egység, amely képes kezelni a saját állapotát (state) és megjeleníteni egy specifikus UI elemet.
* **Virtuális DOM**: A React a virtuális DOM-ot használja, amely egy memóriában tárolt reprezentációja a valódi DOM-nak. Ez lehetővé teszi, hogy a React gyorsabban frissítse a felhasználói felületet, mivel csak azokat az elemeket módosítja, amelyek ténylegesen változtak, ezzel optimalizálva a teljesítményt.
* **Egyirányú adatáramlás (one-way data flow)**: A React alkalmazásokban az adatok csak egy irányban áramlanak: a szülő komponensből a gyermek komponens felé. Ez segít a komponensek állapotának kezelésében és a kód átláthatóságának javításában.
* **JSX szintaxis**: A React lehetővé teszi a **JSX** (JavaScript XML) szintaxis használatát, amely lehetővé teszi, hogy a HTML-t és a JavaScript kódot egyesítve írjunk komponenseket. Bár a JSX nem egy kötelező eleme a React-nak, használata megkönnyíti a kód olvashatóságát és a UI komponensek fejlesztését.
* **React Hooks**: A React 16.8-as verziója óta bevezette a hook-okat, amelyek lehetővé teszik a funkcionális komponensek számára az állapot és egyéb React funkciók (például élettartam-állapotok) használatát, anélkül hogy osztály alapú komponensekre lenne szükség.

### Alapvető fogalmak:

* **Komponens**: Egy React alkalmazás építőeleme. Minden komponens rendelkezhet saját állapottal (state), és képes renderelni HTML-szerű tartalmat.
* **State és Props**:
  + **State**: A komponens belső adatainak tárolására szolgál. A state változtatásával a komponens újra renderelődik.
  + **Props**: A komponens külső adatainak (pl. szülő komponens által átadott információk) kezelésére szolgál. A props változtathatatlanok (immutable), és egyirányú adatáramlással jutnak el a gyermek komponensbe.
* **Lifecycle Methods**: A React komponensek életciklusa során különböző események történnek, mint például a komponens létrehozása, frissítése és törlése. Az osztály alapú komponensek használják ezeket a metódusokat, de funkcionális komponensekben a hook-okkal is elérhetjük az életciklus eseményeket.

**Mysql:**

A **MySQL** egy nyílt forráskódú, relációs adatbázis-kezelő rendszer (RDBMS), amelyet adatbázisok kezelésére, tárolására és lekérdezésére használnak. A MySQL a világ egyik legnépszerűbb adatbázis-kezelő rendszere, és gyakran használják webalkalmazások, vállalati rendszerek, illetve más adatkezelési igények kezelésére.

### Főbb jellemzők:

* **Relációs adatbázis-kezelő rendszer (RDBMS)**: A MySQL adatbázisai táblákban tárolják az adatokat, és az adatok közötti kapcsolatokat kulcsok és idegen kulcsok segítségével modellezhetjük. Az SQL nyelvet használja az adatbázis-kezeléshez.
* **SQL támogatás**: A MySQL SQL nyelvet használ az adatok lekérdezésére és manipulálására. Az SQL parancsokkal lehetőség van adatokat hozzáadni, módosítani, törölni és lekérdezni, valamint bonyolultabb műveletek végrehajtására is, például szűrés, rendezés, összegzés.
* **Magas teljesítmény**: A MySQL optimalizált, és képes nagy mennyiségű adat gyors kezelésére. Emellett számos lehetőség van a teljesítmény javítására, például indexek és cache-ek használatával.
* **Nyílt forráskód**: A MySQL nyílt forráskódú szoftver, ami azt jelenti, hogy bárki ingyenesen használhatja, módosíthatja és terjesztheti.
* **Széles körű támogatás**: A MySQL kompatibilis számos operációs rendszerrel (Windows, Linux, macOS), és rengeteg programozási nyelv (például PHP, Python, Java) támogatja a MySQL adatbázisokkal való munkát.
* **Replikáció és skálázhatóság**: A MySQL képes adatbázis replikációra, ami lehetővé teszi az adatok másolását egyik szerverről a másikra, ezzel javítva a rendelkezésre állást és terhelés elosztást. Emellett a MySQL jól skálázható, tehát képes megbirkózni egyre nagyobb adatbázisokkal és felhasználói terheléssel.

### Alapvető műveletek:

* **Adatok létrehozása**: Az adatokat INSERT INTO utasítással adhatjuk hozzá az adatbázishoz.
* **Adatok lekérdezése**: Az SELECT utasítás segítségével kérhetjük le az adatokat az adatbázisból.
* **Adatok frissítése**: Az UPDATE utasítással módosíthatjuk az adatokat.
* **Adatok törlése**: A DELETE utasítással törölhetjük az adatokat.

A **MySQL** egy nyílt forráskódú, relációs adatbázis-kezelő rendszer (RDBMS), amelyet adatbázisok kezelésére, tárolására és lekérdezésére használnak. A MySQL a világ egyik legnépszerűbb adatbázis-kezelő rendszere, és gyakran használják webalkalmazások, vállalati rendszerek, illetve más adatkezelési igények kezelésére.

### Főbb jellemzők:

* **Relációs adatbázis-kezelő rendszer (RDBMS)**: A MySQL adatbázisai táblákban tárolják az adatokat, és az adatok közötti kapcsolatokat kulcsok és idegen kulcsok (foreign keys) segítségével modellezhetjük. Az SQL (Structured Query Language) nyelvet használja az adatbázis-kezeléshez.
* **SQL támogatás**: A MySQL SQL nyelvet használ az adatok lekérdezésére és manipulálására. Az SQL parancsokkal lehetőség van adatokat hozzáadni, módosítani, törölni és lekérdezni, valamint bonyolultabb műveletek végrehajtására is, például szűrés, rendezés, összegzés.
* **Magas teljesítmény**: A MySQL optimalizált, és képes nagy mennyiségű adat gyors kezelésére. Emellett számos lehetőség van a teljesítmény javítására, például indexek és cache-ek használatával.
* **Nyílt forráskód**: A MySQL nyílt forráskódú szoftver, ami azt jelenti, hogy bárki ingyenesen használhatja, módosíthatja és terjesztheti.
* **Széles körű támogatás**: A MySQL kompatibilis számos operációs rendszerrel (Windows, Linux, macOS), és rengeteg programozási nyelv (például PHP, Python, Java) támogatja a MySQL adatbázisokkal való munkát.
* **Replikáció és skálázhatóság**: A MySQL képes adatbázis replikációra, ami lehetővé teszi az adatok másolását egyik szerverről a másikra, ezzel javítva a rendelkezésre állást és terhelés elosztást. Emellett a MySQL jól skálázható, tehát képes megbirkózni egyre nagyobb adatbázisokkal és felhasználói terheléssel.

### Alapvető műveletek:

* **Adatok létrehozása**: Az adatokat INSERT INTO utasítással adhatjuk hozzá az adatbázishoz.
* **Adatok lekérdezése**: Az SELECT utasítás segítségével kérhetjük le az adatokat az adatbázisból.
* **Adatok frissítése**: Az UPDATE utasítással módosíthatjuk az adatokat.
* **Adatok törlése**: A DELETE utasítással törölhetjük az adatokat.

### Előnyök:

* **Nyílt forráskódú és ingyenes**: A MySQL szabadon hozzáférhető, és számos ingyenes eszköz áll rendelkezésre a használatához.
* **Skálázhatóság és teljesítmény**: Nagy adatbázisok kezelésére is alkalmas, és többféle optimalizációs lehetőséget kínál.
* **Széles körű dokumentáció és közösségi támogatás**: A MySQL rendkívül jól dokumentált, és számos fejlesztői közösség áll rendelkezésre, akik segítenek a problémák megoldásában.
* **Kompatibilitás más rendszerekkel**: Különböző platformokon és programozási nyelvekben használható, széles körben támogatott adatbázis-kezelő.

### Használati területek:

* **Webalkalmazások**: A MySQL gyakran használatos a weboldalak backendjében, ahol adatokat kell tárolni, például felhasználói profilokat, termékadatokat vagy megrendeléseket.
* **Vállalati alkalmazások**: Nagyvállalatok is használják a MySQL-t adatbázis-kezelési feladatokra, például CRM rendszerekhez vagy ERP rendszerekhez.
* **Nyilvános API-k és adatbázisok**: A MySQL gyakran szerepel nyilvános adatbázisok kezelésében, ahol adatokat biztosítanak különböző alkalmazások számára.