JobService

SZAKDOLGOZAT TERVEZET

### Témaválasztás

Azt lehet észrevenni, hogy manapság az embereknek napról napra nehezebb munkát találni, a cégeknek pedig egyre nehezebb munkást találni, mindkét oldalon egyszerre tűnik úgy, mint ha hiány lenne, miközben igény van. Vagyis a valódi probléma inkább az, hogy az álláshirdetések nem elég láthatók, illetve az álláskeresőknek nincs elég lehetőségük láthatóvá tenni magukat.

Ezért az elképzelésünk egy összetett álláshirdető oldal volt, ahol egyaránt lehet hirdetni magukat a munkakeresőknek és a munkavállalóknak, emellett pedig egy adminisztrátor felületet is társítanánk hozzá, ahol ezeket törölni, illetve módosítani lehet abban az esetben, ha az szükséges. Az álláshirdetők nagy mértékben befolyásolhatják, hogy a hirdetésük hogyan jelenlen meg, nem pusztán szöveget, de képeket is enged megjeleníteni a hirdetés szövege közé szúrva, emellett kulcsszavakat adhat a hirdetéséhez, amivel az álláskeresők könnyebben meg tudják találni, ha erre keresnek. Az álláskeresők szintén rengeteg eszközt kapnak hirdetésük személyre szabásához. Mindkét félnek módjában áll extra költségek fejében „kiemelt” hirdetést közzé tennie, amik rend szerint a hirdetések első oldalain lesznek beszúrva az átlagos hirdetések közé. Az adminisztrátor feladata a hamis hirdetések eltávolítása, legyen az állás vagy álláskereső, esetlegesen hibásan feladott hirdetések módosítása, ha olyan jellegű a probléma, amit a felhasználó maga nem tud kijavítani, illetve egyéb karbantartási feladatok.

# **Használt technológiák, környezetek, nyelvek**

## REACT

A React (vagy React JS) egy olyan JavaScript nyelven írt nyílt forráskódú könyvtár, ami elősegíti a felhasználói felületek programozását, elsősorban webes alkalmazások létrehozását. A könyvtár alapú megközelítés lehetővé teszi, hogy a fejlesztő teljes szabadsággal rendelkezzen a fejlesztésre használt technológiák kiválasztásában. Vagyis eldöntheti, hogy a React könyvtárból csupán egyetlen komponenst hív segítségül, vagy hogy akár a React legyen a fejlesztői környezetének súlypontja. Szemben egy keretrendszerrel, ami nagymértékben meghatározza és előírja a fejlesztés szabályait, kereteit.

A React előnyei:

- megkönnyíti komponensek írását;

- újra felhasználható komponensekkel növeli a hatékonyságot, átláthatóságot

- keresőmotor-barát forráskód megjelenítés

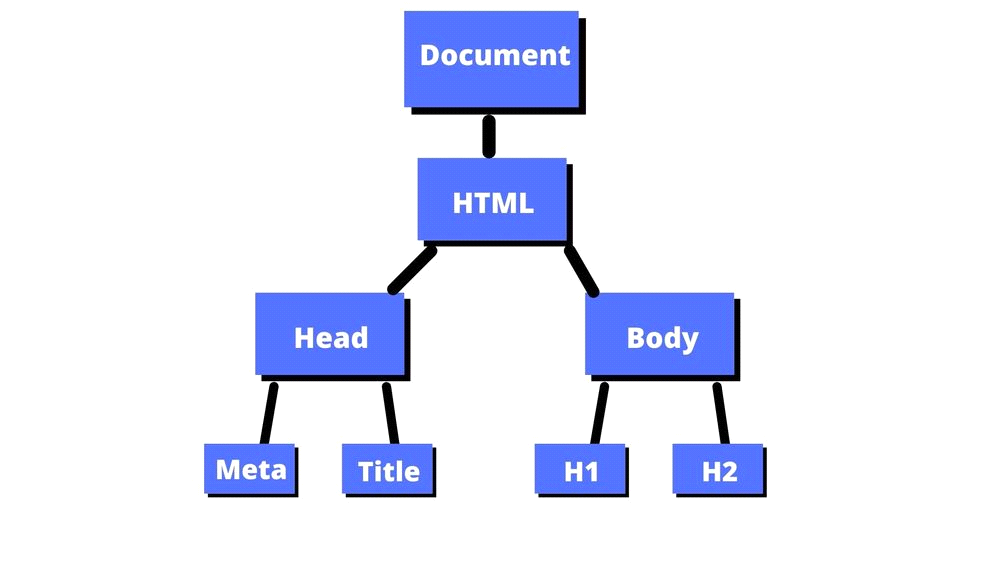
- magas népszerűsége miatt széleskörű fejlesztési támogatás

- további tökéletesítése várható a facebook-nak köszönhetően

A számítógépes alkalmazáshoz egyik fontos tulajdonsága az interaktivitás. A webes alkalmazások ezt jellemzően a böngészőben megjelenő HTML elemek közvetlen módosításával érik el.

#### DOM [1]

A böngésző ablakát képező objektum a "document", ezen jelenik meg az összes HTML elem, az ezek felépítését leíró modell, a DOM (Document Object Model) ennek köszönheti a nevét. A HTML elemek jellemzően egymásba ágyazódnak, egy fához hasonló szerkezetet képezve.



A HTML elemek a tartalmat jelenítik meg, a CSS stílusok a megjelenés módját határozzák meg, az ilyen tartalmat és stílust pedig a JavaScript segítségével tudjuk elérni, módosítani.

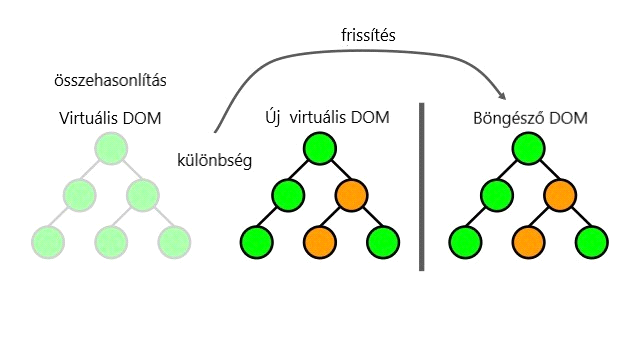
A React előtti technológiák a böngészőben megjelenő elemeket közvetlenül manipulálták a DOM segítségével. Például egy felhasználó bejelentkezésekor egy JavaScript kód a böngésző jobb felső sarkában megjelenítette a felhasználó nevét, a bal oldalon behívta a felhasználó profilját, egy másik oszlopban megjelenítette a barátait is, majd a másik oldalon egy csevegő ablakot is fel kellett dobnia, és felette még ki kellett írnia, hogy hány új üzenetet és értesítést kapott.

A megírt kód a különböző böngészőkben néha eltérően viselkedett, és jelentős programozási ráfordítást jelentett, hogy egy weboldal a legtöbb böngészőben azonos módon jelenjen meg. A jQuery megjelenése erre adott választ, és nemcsak a böngészők közötti kompatibilitást segítette elő, hanem kis JavaScript könyvtárként a HTML elemek elérését és módosítását is egyszerűbbé tette. A fejlesztők idővel egyre nagyobb alkalmazásokat hoztak létre, ahol a JavaScript és az interaktivitás növekvő szerepet kapott. Megjelentek a JavaScript könyvtárak, amelyek az egyre több JS funkciót és fájlt rendszerbe szervezik. A felhasználói élmény növelése, a betöltési idők csökkentése céljából létrejöttek az egyoldalas alkalmazások (SPA), és aszinkron kommunikációt elősegítő technológiák, mint például az AJAX. 2010 körül a Google által kifejlesztett Angular JS keretrendszer vált a webes alkalmazások fejlesztésének egyik meghatározó eszközévé. Ez már valóban nagy és összetett alkalmazások létrehozását tett lehetővé. A HTML, CSS és JavaScript kód továbbra is jellemzően elkülönült, a növekvő interaktivitás miatt ezek összekapcsolása és kezelése egyre több JavaScript kód megírását igényelte. A DOM manipulálása egy összetett alkalmazásban egyre nehezebbé vált, hiszen egyetlen esemény akár több tucatnyi reakciót válthat ki, amelyek szintén további reakciókat eredményezhetnek. Ez megnehezíti az események és a hozzátartozó esetek átláthatóságát, karbantartását.

A DOM manipuláció erőforrásigényes feladat: a böngésző megjeleníti a módosított elemet, és ekkor az oldal elrendezését is gyakran módosítania kell. Rádásul néhány JS keretrendszer felesleges DOM frissítéseket is végez, például egy hosszú lista egyetlen elemének módosulásakor akár a teljes listát újra építheti. A React alkotói erre a problémára hatékony megoldást adtak.

#### Virtuális DOM [2]

A Reactben egy DOM objektumhoz egy virtuális DOM objektum kapcsolódik. Ezt a DOM objektum leegyszerűsített másolataként képzelhetjük el, ami nem látható a böngészőben. Ez hátránynak tűnik, de éppen ebből adódnak az előnyei. A módosítása ezáltal sokkal gyorsabb és egyszerűbb, hiszen ilyenkor semmi nem jelenik meg a képernyőn, a tényleges DOM nem módosul. A virtuális DOM változása esetén a React kizárólag azokat a DOM objektumokat frissíti, amelyek virtuális megfelelői módosultak, ami jelentősen jobb teljesítményt eredményez a többi hasonló technológiához képest.



##### MySQL [3]

A MySQL egy relációs adatbázis-kezelő rendszer. A relációs adatbázisok külön táblákban tárolják az adatokat, ahelyett, hogy az összes adatot egyetlen nagy tárhelybe helyeznék el. A logikai modell az adatbázisok, táblák, nézetek, sorok és oszlopok, mint objektumok révén rugalmas programozási környezetet kínál. Az adatmezőkhöz számos adattípus közül választhatunk, és a mezők közötti kapcsolatokat is szabályozhatjuk attól függően, hogy egy az egyhez (1:1), egy a többhöz (1: N) vagy több a többhöz (N:M) kapcsolatról van szó. Ezen kívül további tulajdonságokat rendelhetünk az egyes mezőkhöz (pl. egyedi, kötelező vagy nem kötelező, automatikus növelés stb.). Az adatbázis érvényesíti ezeket a szabályokat, így egy jól megtervezett adatbázis esetén az alkalmazás soha nem lát inkonzisztens, duplikált, árva, elavult vagy hiányzó adatokat.

A MySQL név SQL része a „Structured Query Language” rövidítése. Az SQL az adatbázisok elérésére használt legelterjedtebb szabványosított nyelv. A programozási környezettől függően közvetlenül futathatjuk az SQL utasításokat (például jelentések létrehozásához), egy másik nyelven írt kódba is beágyazhatjuk, vagy nyelvspecifikus API-t használhatunk, amely elrejti az SQL szintaxist.

A MySQL adatbázis szerver nagyon gyors, megbízható, méretezhető és könnyen használható. A MySQL szerver kényelmesen futhat asztali számítógépen vagy laptopon, más alkalmazások, webszerverek mellett, kevés figyelmet igényel. Eredetileg nagy adatbázisok kezelésére fejlesztették ki, és évek óta sikeresen használják nagy igénybevételt jelentő éles környezetben. A funkciók gazdag és hasznos készletét kínálja, miközben folyamatos fejlesztés alatt áll. Kapcsolódása, sebessége és biztonsága miatt a MySQL szerver kiválóan alkalmas az internetes adatbázisok elérésére.

###### Az adatbázis szerkezete [4]

Az adatbázis az adatok és a köztük lévő összefüggések rendszere, amelyet egymás mellett tárolunk. A relációs adatmodellben az adatokat egymással logikai kapcsolatban álló táblákba rendszerezzük. Ennek során követelmény, hogy minden oszlopnak egyértelmű neve van, minden sorban ugyanazok az oszlopok vannak, az oszlopokban található adatok meghatározott értéket vehetnek fel, az oszlopok soronként csak egy értéket vehetnek fel, és a táblát a neve egyértelműen azonosítja.

A tábla a logikailag összetartozó adatokat foglalja össze. A tábla oszlopokból és sorokból áll, amelyeket mezőknek, illetve rekordoknak nevezünk. A rekord az adatbázis egy sora. Egy rekordban tároljuk az egymással összefüggő adatokat. Ezzel szemben a mező az adatbázis egy oszlopa, amelyben az egyedek tulajdonságértékeit tároljuk.

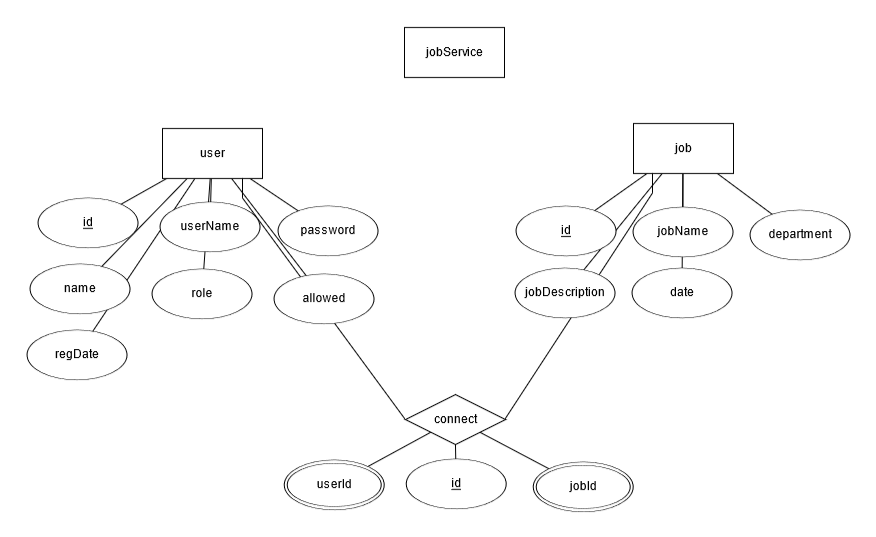
Az egyed az, amit le akarunk írni, amelynek az adatait tároljuk és gyűjtjük az adatbázisban. Az egyedet idegen szóval entitásnak nevezzük. Egyednek tekinthetünk például egy személyt. Az attribútum (vagyis tulajdonság) az egyed valamely jellemzője. Az egyed az attribútumok összességével jellemezhető. Egy személy egy jellemzője lehet például a neve. Az egyedre vonatkozóan megadott tulajdonságok összességét egyedtípusnak nevezzük. Az egyedre vonatkozóan megadott konkrét tulajdonságokat egyed-előfordulásnak nevezzük. Egy egyed-előfordulás például Kis Ede, aki 29 éves, 183 cm magas, kék szemű, barna hajú.

Elsődleges kulcsnak hívjuk a a tábla rekordjainak egyértelmű azonosítóját, ami egyedi értékekkel rendelkezik. Az idegen kulcs olyan azonosító, amelynek segítségével egy másik tábla elsődleges kulcsára hivatkozhatunk.

## Adatbázis

A munkaközvetítő alkalmazásunk adatbázisának tervezésekor hatékonysági szempontokat is figyelembe vettünk, ennek fényében a lehető legátláthatóbb szerkezet kialakítására törekedtünk. Elvileg lehetséges lett volna külön tábla létrehozása mind a munkaadók, mint a munkakeresők számára. Azonban több szempontból is optimálisabb megoldásnak tűnt, ha az összes felhasználót egy táblában tartjuk nyilván, és szerepkörökkel különböztetjük meg őket. Ezáltal további típusú felhasználó is könnyen definiálható, mint például az adminisztrátor.

Az adatbázisunk kezdeti EK diagramja [5] a következő:



## WCF [6]

A WCF, vagyis Windows Communication Foundation egy alapvetően szolgáltatás orientált alkalmazásokhoz készült keretrendszer. A segítségével adatokat küldhetsz egyik végponttól a másikba, legyen az egy kliens oldali alkalmazás, vagy egy másik szerver oldali szolgáltatás. Az üzenetek egyszerű karakterektől kezdve XML, JSON, illetve bináris adatfolyamokig bármilyen komplexitásúak lehetnek. A WCF tehát használható üzleti tranzakciók feldolgozására, időjárás, úti forgalom adatok továbbítására vagy akár egy felhasználók közötti üzenetküldő alkalmazásra, és még sok másra. Ezeket természetesen a WCF megjelenése előtt is meg lehetett csinálni, de a WCF sokkal könnyebbé teszi a végpontok közötti kommunikációt.

A szolgáltatás orientált keretrendszerek előnye, hogy a kiszolgált végpontokkal elég lazán kapcsolatban lenniük és nem kell semmit vagy szinte semmit változtathatatlanul belekódolni a kliens alkalmazásunkba, mint például egy előre meghatározott cím, ahol el lehet érni a szolgáltatást (pl. http://localhost:3000/), akármi is változik annak kódjában, az egyetlen változtathatatlan dolog ez az IP cím lesz.

Az üzenetváltásra többféle séma is van. A leggyakoribb megoldás a kérés/válasz séma, ahol az egyik végpont adatot kér a második végponttól, a második végpont pedig erre válaszol a kérésnek megfelelően. De van egy irányú séma is, amely esetben egy végpont küld adatot a másiknak anélkül, hogy az válaszra várna ez után. Egy komplexebb séma a két irányú csere, ahol két végpont kapcsolatot létesít, és oda visszaküld adatokat hasonlóan egy üzenetváltó alkalmazáshoz.

A WCF támogatja a szolgáltatás metaadatának kiadását ipari szabványoknak megfelelő formátumokban, mint WSDL, XML vagy WS-Policy. Ezek a metaadatok felhasználhatóak arra, hogy automatikusan előállítsunk és konfiguráljunk klienseket, amik elérhetik a szolgáltatásunkat. http és HTTPS, illetve Web Service Metadata Exchange szabványoknak megfelelve egyaránt kiadható ez a metaadat.

Az üzeneteket titkosítani lehet a bizalmas adatok védelme érdekében, ez esetben a felhasználónak hitelesítenie kell magukat, hogy ezeket az üzeneteket megkaphassak. A titkosítás megoldható több jól ismert szabvánnyal, mint az SSL vagy a WS-SecureConversation.

Az üzeneteket több beépített protokollon és kódoláson keresztül. A leggyakoribb protokoll és kódolás séma kódolt szöveges SOAP üzeneteket küldeni http (HyperText Transfer Protocol) segítségével Interneten keresztüli használatra. A WCF továbbá megengedi a TCP, nevesített csövek, illetve MSMQ segítségével való üzenetküldést is. Az üzenetek vagy szövegként, vagy optimalizált bináris formában kódolva küldhetők. Bináris adatok esetén az MTOM szabvány segítségével hatékonyan küldhetünk adatot. De abban az esetben, ha egyik formátum sem felel meg az elvárásainknak, elkészíthetjük a saját szállítás vagy kódolás formátumunkat is.

A tartós üzenetek olyan üzenetek, melyek soha nem vesznek el valamilyen kommunikációs probléma fellépése esetén. Az üzenetek egy ilyen sémában mindig elmentődnek egy adatbázisba, így abban az esetben, ha valami oknál fogva megszakad a kapcsolat, az adatbázis lehetővé teszi, hogy folytassuk az üzenetváltást onnan, ahol a szolgáltatás megszakadt. A Windows Workflow Foundation segítségével szintén létrehozhatók ilyen tartós üzenetek.

A REST egy példája a fejlődő Web 2.0 technológiáknak. A WCF konfigurálható, hogy fel tudjon dolgozni „egyszerű” XML adatot, ami nincs SOAP csomagba csomagolva. A WCF továbbá kibővíthető, hogy egy egy kifejezett XML formátummal dolgozzon, mint az ATOM, ami egy népszerű RSS szabvány, vagy egyéb nem XML formátumokba, mint a Javascript Object Notation, vagyis JSON.

A WCF egy nagyon rugalmas felület, ebből kifolyólag rengeteg más Microsoft termékkel használható. Az egyik ilyen például a Windows Workflow Foundation(WF). A munkamenetek leegyszerűsítik az alkalmazásfejlesztést azzal, hogy a munkamenet egyes lépéseit úgynevezett „aktivitásokba” foglalják. Az első verzióban még ugyan nem, de az azt követőben beépített része lett a WCF. Ez lehetővé tette, hogy bármilyen munkamenetet könnyedén ki tudjunk szolgálni egy WCF szolgáltatással. További termékek közé tartozik még többek között a:

* Microsoft BizTalk Server R2:

- egy kommunikációs technológia, ami egy szabvány formátumból alakít át adatot egy másikba   
- WCF szolgáltatást használva tud adatokat továbbítani a központi tárolójába, ahol ezek az adatok átalakítódnak

* Microsoft Silverlight:

- média-igényes weboldalak készítésére készült, mint például egy videó streaming szolgáltatás  
- a második verziótól kezdődően a WCF egy beépített része lett a technológiának, ahol a kliens/szerver kommunikáció elvégzésére használható

* Windows Server AppFabric:
* kifejezetten arra lett felépítve, hogy WCF kommunikációt használó alkalmazásokat vezéreljen  
  - kifejezetten WCF alkalmazásokra tervezett konfigurációs beállításokkal és terjedelmes vezérlési lehetőségekkel van ellátva

További fejlesztési lehetőségek

Összefoglalás

Forrásjegyzék

#### [1] <https://www.freecodecamp.org/news/what-is-the-dom-document-object-model-meaning-in-javascript/>, felkeresés időpontja: 2021.12.14.

[2] <https://javascript.plainenglish.io/react-the-virtual-dom-comprehensive-guide-acd19c5e327a>, felkeresés időpontja: 2021.12.14.

[3]<https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html>, felkeresés időpontja: 2021.12.14.

[4] <https://arato.inf.unideb.hu/bujdoso.gyongyi/kurzusok/adatbazis/ea/adb_1ea_1_Alapok_2016.pdf>, felkeresés időpontja: 2021.12.14.

[5] Egyed-Kapcsolat (EK) diagramkészítő felület forrása: <https://erdplus.com/standalone>, felkeresés időpontja: 2021.12.14.

[6] <https://docs.microsoft.com/en-us/dotnet/framework/wcf/whats-wcf>, felkeresés időpontja: 2021.12.14.