**Miskolci Egyetem**



Gépészmérnöki és Informatikai Kar

Informatikai Tanszék

Taraszovics Károly

**E-learning webalkalmazás fejlesztése**

Konzulens:

Smid László

Miskolc, 2019

**Eredetiség Nyilatkozat**

Alulírott Taraszovics Károly; Neptun-kód: EDXK4L

a Miskolci Egyetem Gépészmérnöki és Informatikai Karának végzős mérnökinformatika szakos hallgatója ezennel büntetőjogi és fegyelmi felelősségem tudatában nyilatkozom és aláírásommal igazolom, hogy

*E-learning webalkalmazás fejlesztése*

című szakdolgozatom saját, önálló munkám; az abban hivatkozott szakirodalom

felhasználása a forráskezelés szabályai szerint történt.

Tudomásul veszem, hogy szakdolgozat esetén plágiumnak számít:

* szószerinti idézet közlése idézőjel és hivatkozás megjelölése nélkül;
* tartalmi idézet hivatkozás megjelölése nélkül;
* más publikált gondolatainak saját gondolatként való feltüntetése.

Alulírott kijelentem, hogy a plágium fogalmát megismertem, és tudomásul veszem, hogy

plágium esetén szakdolgozatom visszautasításra kerül.

Miskolc,2019. április 28.

…….……………………………….…

Hallgató

# Tartalomjegyzék

[Tartalomjegyzék 1](#_Toc21008985)

[Abstract 2](#_Toc21008986)

[1. Bevezetés 3](#_Toc21008987)

[2. Felhasznált technológiák 6](#_Toc21008988)

[2.1 Java 7](#_Toc21008989)

[2.2 Apache Tomcat Webserver 8](#_Toc21008990)

[2.3 HTML, CSS, JavaScript 9](#_Toc21008991)

[2.4 AJAX 11](#_Toc21008992)

[2.5 JSON 12](#_Toc21008993)

[2.6 PostgreSQL 13](#_Toc21008994)

# Abstract

**Az internet a gyors és naprakész információszerzés, a hatékony kapcsolatteremtés és a tartalmas szórakozás mellett a tanulásnak is kiváló közege. Az interneten rengeteg oktató jellegű és ismeretterjesztő oldal található, de bővelkedik az online tananyagokban is. Az interneten keresztüli tanulást/oktatást összefoglaló néven e-learningnek nevezzük.**

**Napjainkban egyre gyakoribbak és népszerűbbek azok a weboldalak, amik segítségével számos különböző témában, különböző online tananyagokhoz tudunk hozzáférni.**

**Szakdolgozatom témája egy webalkalmazás készítése, amely lehetőséget nyújt a felhasználóknak saját fiókot létrehozni, bejelentkezni a szükséges autentikációs folyamat sikeres lezárulta után, böngészni különböző témákban, különböző tananyagokban, valamint meg tudják vásárolni és meg tudják tekinteni azokat.**

# Bevezetés

A webalkalmazások már azelőtt jelen voltak, mielőtt az internet ekkora népszerűségre tett volna szert. Például Larry Wall 1987-ben feltalálta a Perl-t, ami egy szerver-oldali scriptnyelv. Ez hét évvel azelőtt volt, hogy az internet elterjedt volna az akadémiai és tudományos körökön kívül. Az első alap webalkalmazások még relatíve egyszerűen voltak, de a kilencvenes évek végétől már egyre komolyabb és komplexebb alkalmazások születtek. Napjainkban már emberek milliói használnak webalkalmazásokat jövedelmük online aktázására, online banki tranzakciók lebonyolítására, kapcsolattartásra családi és baráti körben, és még számtalan más célra.

Szakdolgozatom célja egy webalkalmazás (más néven webapplikáció) készítése. A webalkalmazás egy komplex számítógépes program, amely webböngészőt és különböző webfejlesztési technológiákat használ, hogy feladatokat hajtson végre az interneten.

Az alkalmazásnak szüksége van egy hátsó rétegre, amely az alkalmazás logikájának és funkcióinak megvalósításáért felelős, így képessé téve az alkalmazást a kliens-oldali kérésekre való reagálásra. Ezt a hátsó réteget backendnek, vagy szerver-oldalnak nevezzük.

Az alkalmazás másik fontos része a kliens-oldal, vagy más néven frontend. Ez az a réteg, amivel az alkalmazás felhasználói interakcióba lépnek. Ez a réteg felelős a webalkalmazás megjelenéséért. Betűtípustól és betűszíntől kezdve, legördülő menüpontokon át egészen a hátterekig, minden megjelenési részletet a frontend valósít meg, felhasználva HTML, CSS és JavaScript elemeket a böngésző által kontrollálva.

A HTML egy angol mozaikszó, jelentése HyperText Markup Language. vagyis hiperszöveges jelölőnyelv. A hipertext az interneten található oldalakat (dokumentumokat) jelenti, amely szöveg, kép, videó, hang, vagy egyéb médiaelemek kombinációjából áll. A HTML ezeknek az oldalaknak a formázását, elrendezését tartalmazza saját kódnyelvén.

A CSS a Cascading Style Sheets angol kifejezés rövidítése. Jelentése egymásba ágyazott stíluslapok. Ez a HTML dokumentum megjelenéséért felelős egyszerű nyelv, mely segítségével kontrollálhatjuk, hogy hogyan és hol jelenjelek meg a dokumentum egyes paragrafusai, címsorai, képei, stb. Az egymásba ágyazhatóság arra utal, hogy több stílust is meghatározhatunk, valamint egy stílus több elemre is érvényes lehet.

A JavaScript egy, a webfejlesztésben használt programnyelv, amely azt teszi lehetővé, hogy a weboldalak anélkül reagáljanak felhasználói kérésekre, hogy egy új oldal töltődne be. Az imént felsorolt három webfejlesztési technológia segítségével már lehetséges működő, komplex webalkalmazásokat fejleszteni, holott létezik még számtalan keretrendszer, amelyek kibővítik a webfejlesztés lehetőségeit. Ilyenek például a Bootstrap, AngularJS, ReactJS, Foundation, EmberJS, Backbone, vagy a Vue.js.

Habár a Java nyelvet az egyik legbonyolultabb és legkomplexebb nyelvként tartják számot, mégis talán ez a legszélesebb körben használt nyelv backend oldal fejlesztésére, kiegészítve különböző keretrendszerekkel, mint a Spring vagy Java EE. A Java nyelven kívül természetesen számos más backend nyelv is használatos, többek között C#, PHP, Ruby vagy Pyhton.

Az alkalmazás fő célja, hogy használható legyen a kliensek számára a böngészőben.

A fő funkciók közé tartozik a felhasználók regisztrációja és bejelentkezése. Az alkalmazás különböző szerepkörökhöz különböző jogosultságokat biztosít. A felhasználók különböző témájú tananyagok között böngészhetnek, pénzt tölthetnek fel a fiókjukba, megvásárolhatják, megtekinthetik a tananyagokat, valamint információt láthatnak a saját fiókjukról és számon tarthatják, milyen már megvásárolt tananyagaik vannak. Az oldal kezelője (admin) új tananyagokat tud hozzáadni a rendszerhez, és hozzáfér az összes már létező tananyaghoz.

# Felhasznált technológiák

Egy webalkalmazás fejlesztésénél el kell döntenünk, hogy az single-page webalkalmazás, vagy multi-page webalkalmazás lesz-e. A multi-page webalkalmazás nevéből is kikövetkeztethető, hogy több oldalból áll. Minden felhasználói interakciónál újra be kell tölteni az egész oldalt, akármilyen kicsi is legyen a változtatni kívánt elem. Az ilyen technológiát például akkor használjuk, ha hatékony keresőmotor-optimalizálásra van szükségünk, mivel egy oldalhoz egy keresőszót rendelhetünk hozzá.

A single-page webalkalmazás napjaink elterjedtebb formája, gyorsasága miatt. A legtöbb forrás csak egyszer töltődik be egy alkalmazás futtatásakor, felhasználói interakcióknál pedig az oldal dinamikusan frissül, ami azt jelenti, hogy az oldalnak csak az éppen megváltozó elemeit kell betölteni. Nem kell messze mennünk, ha példát akarunk keresni single-page webalkalmazásokra, ide sorolhatjuk például a Gmail-t a Google Maps-ot, a Facebook-ot, vagy akár a GitHub-ot is.

Az én alkalmazásom egy single-page alkalmazás, melynek fejlesztése során nem csak a bevezetőben említett három technológiát használtam. Lássuk tehát, miket még!

# Java

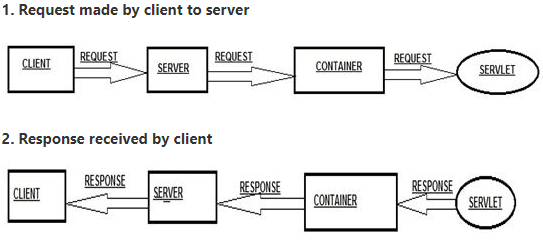
James Gosling, aki feltalálta a Java nyelvet, nem számított arra, hogy azt ilyen széles körben fogják majd használni. 1996-ban az internet még csak egy újdonság volt, Gosling így a Javát kisebb eszközök kommunikálásának megvalósítására tervezte.

A Java API fejlődésében a Java 2 nagy mérföldkőnek számított. A Java 5. kiadásában kerültek bemutatásra a generikusok, ami hatalmas változást hozott a Java nyelv szintaxisában. A Google 2009-ben kiadta az Android ’Software Developement Kit’-et (SDK), ami Java nyelven íródott. Ez szintén mérföldkő volt a nyelv történetében. Napjainkra, a Java a legnépszerűbb objektum orientált programnyelvvé fejlődött, legutóbbi kiadása, a Java 8 2014-ben jelent meg.

Alkalmazásom backend rétege is Java nyelven íródott. Egyik oka, hogy választásom a Java-ra esett, egy angol kifejezéssel fejezhető ki: „Write once, run everywhere”, ami annyit tesz, hogy bármilyen eszközön, amelyen telepítve van a JVM (Java Virtual Machine), futtatható egy lefordított Java program. A másik meghatározó ok, ami miatt a Java mellett tettem le a voksomat, a Java memóriakezelésének előnyei. A Java automatikus memória menedzselése növeli a webalkalmazások hatékonyságát és gyorsaságát. Kétféle memóriaterületet különböztethetünk meg, a heap-et és a stack-et. A példányváltozók és a példányosított objektumok a heap-en helyezkednek el, míg a metódusokban definiált lokális változók a stack-en. A Java szemétgyűjtő mechanizmusa, a Garbage Collector (GC) a következőképpen működik: az élő szálak (thread-ek) stack-jein az elérhető objektumok referenciái találhatóak. A Garbage Collector végigjárja ezeket az objektumokat, és megjelöli azokat, amelyek nem érhetők el referenciákon keresztül egyik élő thread-ből sem, majd kidobja a megjelölteket.

# 2.2 Apache Tomcat Webserver

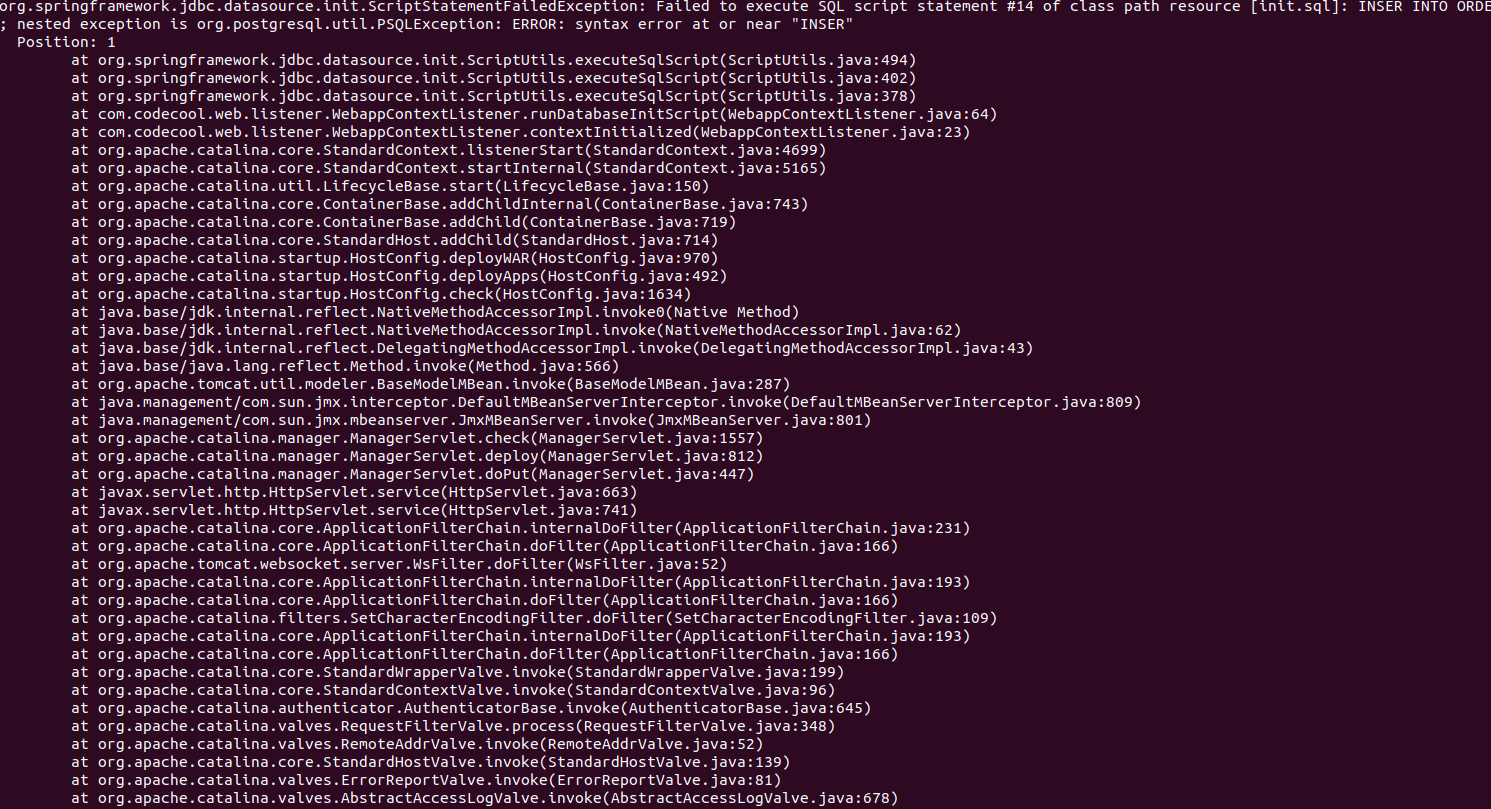
Az Apahe Tomcat Webserver szoftver egy nyílt forráskódú implementáiója a Java Servlet és JavaServer Pages(JSP) specifikációkat. Az ilyen specifikációkat támogató webszervereket servlet containernek nevezzük. Mielőtt a web container fogalmát kifejtenénk, ejtsük nlhány szót a servletekről. A servlet egy olyan Java objektum, amely http kéréseket (request) dolgoz fel és http válaszokat generál (response). A web container egy Java applikáció, ami a servleteket kontrollálja. Amikor a böngészőből (kliens) egy kérés érkezik a webszerver felé. a szerver a container-nek küldi azt, ahelyett, hogy egyenesen a servletnek küldené. A container megkeresi a megfelelő servletet, majd továbbítja a request-et hozzá, majd a servlet, miután legenerálta a megfelelő választ, visszaküldi azt a containernek, hogy az visszaküldje ezt a response-t a server felé. Ezt a folyamatot a következő ábra szemlélteti.



Az alkalmazás futásához tehát futtatni kell egy web container-nek is, amit kezeli a servleteket. Linux terminálból a Tomcat web containert a következő paranccsal indíthatjuk el:



A fejlesztés során a Tomcat számos hasznos információt szolgáltat, többet között a kódban vétett hibákról is üzenetet küld. A következő ábra azt szemlélteti, ahogy a Tomcat mutatja a teljes stack trace-t, vagyis azt, hogy a hiba a kód melyik részeiben jelentkezett:



# 2.3 HTML, CSS, JavaScript

**HTML**

Mint ahogy minden weboldalnak, így az én alkalmazásomnak is alapja a HTML, angolul HyperText Markup Language. Ez a leíró nyelv adja meg, hogy mi szerepeljen a weboldalon, bekezdések, szövegdobozok, legördülő menük és még sorolhatnám. A HTML legaktuálisabb változata a HTML5, aminek fő előnye, hogy a webalkalmazásokhoz ennek használatával nincs szükség különböző pluginek használatához (pl. Adobe Flash, Silverlight stb.). Az alkalmazásban mégsem található egyetlen .html kiterjesztésű fálj sem. Ennek oka, hogy .html fájl helyett egy .jsp kiterjesztésű fájlban található meg a DOM (Document Object Model), amely tartalmazza alkalmazásom elemeit.

**JSP – Java Server Pages**

A JSP egy XML alapú technológia, ami egy létező objektum modell alapján dinamikusan tud webes tartalmat generálni. Szintaxisa hasonló a HTML-hez. Az összes .jsp kiterjesztésű fájl egy servletté fordul le, így vehetjük a JSP-t egy servlet technológiának.

**JavaScript**

Az weboldal összes komponense tehát ebben az egyetlen .jsp fájlban található meg, azok a komponensek is, amelyet éppen nem látszódnak a felhasználói felületen. A technológia, amellyel ez elérhető, a JavaScript. A JavaScript egy szkriptnyelv, amely segítségével komplex dolgokat is megvalósíthatunk egy weboldalon. Ahányszor a weboldal többet csinál, mint egy helyben áll és statikus információt jelenít meg – legyen az időfüggő tartalom megjelenítése, interaktív térképek, animált 2D/3D grafikák, stb. – biztosak lehetünk benne, hogy a JavaScriptnek is köze van a dologhoz. Ha a webes technológiákat egy tortában akarnánk ábrázolni, a JavaScript lenne a harmadik réteg a tortánkban, a HTML és a CSS alatt.

**CSS**

A CSS az angol Cascading Style Sheets, vagyis egymásba ágyazott stíluslapok kifejezés rövidítése. Ha egyszerűen akarnánk megmagyarázni, mire is való, akkor azt mondhatnánk, hogy a HTML leírja, hogy Mit, a JavaScript hogy Mikor, a CSS pedig hogy Hogyan szeretnénk megjeleníteni tartalmat weboldalunkon. A .css kiterjesztésű fájlban a dokumentumunk minden elemére hivatkozhatunk, és megadhatjuk annak stílusát, legyen az szín, betűtípus, betűméret, elhelyezkedés stb. A css szelektorok segítségével tudjuk meghatározni, hogy a dokumentum mely elemére szeretnénk egy stílust megvalósítani. Ha például egy bekezdésre szeretnénk hivatkozni, akkor a css fájlba elég a *p* elemet megjelölni, utána pedig {} szimbólumok közé írhatjuk a mintát, amit a bekezdésre szeretnénk szabni. Ha egy elemet, aminek egyedi azonosítója(id) van paraméterlistájába, akkor a *#idnév* szelektort kell használnunk. Ha egy elemnek class attribútumára akarunk hivatkozni, akkor azt a *.classnév* szelektorral tesszük.

# 2.4 AJAX

Az AJAX is egy angol mozaikszó, az Asynchronous JavaScript and XML, vagyis aszinkron JavaScript és XML kifejezés rövidítése. Ez egy webes technológia, amely használatával gyorsabb, interaktívabb webalkalmazásokat hozhatunk létre az XML, HTML, CSS és JavaScript segítségével. Ha egy példával szeretném szemléltetni, mire is használjuk, amikor például a bejelentkezés gombra kattintunk, JavaScript segítségével egy kérés irányul a szerver felé a begépelt bejelentkezési adatokkal, a szerver ellenőrzi, hogy megfelelő-e a beírt felhasználónév és jelszó, és annak megfelelően frissíti az aktuális képernyőt. Az XML (Extensible Markup Language) egy leíró nyelv, amellyel adatokat tudunk küldeni, gyakran ez használatos webalkalmazásoknál, habár, esetünkben erre a célra a JSON technológiát használom (JSON-ról a későbbiekben). Az AJAX leglényegesebb jellemzője, hogy aszinkron, így a felhasználók tovább használhatják az alkalmazást, miközben a kliens információt kér a szervertől a háttérben.

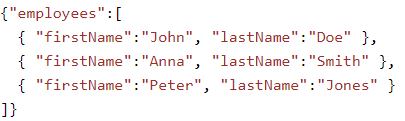
# 2.5 JSON

A JavaScript Object Notation egy kis méretű, szöveg alapú szabvány, amely a JavaScript nyelvből alakult ki. Egy JSON objektumban tárolhatunk mindenféle információt, amit a szerver felé továbbítani akarunk. Minden adat ami AJAX segítségével frissítve lett, eltárolható JSON objektumban a webszerveren. Ezeket a JSON objektumokat amikor szükség van a bennük tárolt adatokra lefordítjuk a JavaScript által értelmezhető nyelvre, és vagy változókba mentjük értéküket, hogy megjelenítés előtt további műveleteket végezhessünk velük, vagy azonnal megfeleltetjük értéküket dokumentumunk valamely elemével, hogy megjelenítsük azt. Mindezt XML segítségével is megtehetnénk, nézzük tehát, miért is választottam a JSON formátum használatát:

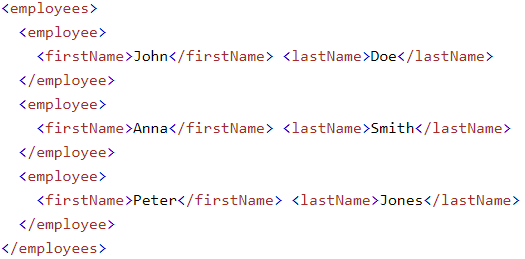
* a JSON formátum rövidebb, mint egy XML
* a JSON-ben használhatunk tömböket
* a JSON rövidebb, könnyebben olvasható és könnyeben írható.

Lássunk egy-egy példát JSON és XML formátumban, ahol mindkét technológia ugyanazt az objektumot modellezi le:

JSON:



XML:



A fentebbi két példa szemmel láthatóan bizonyítja, hogy a JSON formátum sokkal rövidebb és könnyebben olvasható.

# 2.6 PostgreSQL

A PostgreSQL egy erős, nyílt forráskódú relációs adatbáziskezelő rendszer (RDBMS) több, mint 30 év aktív fejlesztői múlttal. Jó hírnevét a megbízhatóság, jó teljesítmény és funckió robosztusság alapozza meg. Eredetileg UNIX-alapú platformokra tervezték, de ma már többféle rendszeren is használatos, többek között Mac OS-en, Solarison vagy Windowson. A PostgreSQL adatbázisok kezeléséhez, lekérdezések írásához a pgAdmin grafikus szoftvert használjuk, amelynek használata egyszerű, és rengeteg lehetőséget nyújt az adatok kezeléséhez. Alkalmazásomban én is a PostgreSQL-t használom.

A pgAdmin-ban létrehoztam egy üres adatbázist, amit egy általam előre megírt inícializációs szkriptfájlból töltök fel. A szkriptfájl kiterjesztése .sql, itt vannak létrehozva a táblák és kapcsolataik valamint itt is vannak feltöltve adatokkal.