Miskolci Egyetem

Gépészmérnöki és Informatikai Kar

Általános Informatikai Intézeti Tanszék



**Ételajánló webes alkalmazás**

**Szakdolgozat**

**Készítette:**

**Név:** Szilágyi Gábor

**Neptunkód:** S9NJK6

**Szak:** Mérnök Informatikus BSc

Korszerű web technológiák szakirány

**1. Bevezetés**

A mindennapjaink egyik fontos része az evés és az ezt megelőző döntéshozatal. A mai választási lehetőségekben, egyre több emberrel esik meg, hogy nem tudja eldönteni mit egyen. Ezen a helyzeten az sem segít, hogy az embereknek egyre kevesebb idejük van arra, hogy saját maguknak főzzenek.

Szakdolgozatom célja olyan webalkalmazás elkészítése, ami ezt a mindennapi döntéshozatalt és a velejáró extra munkát könnyíti meg. Az alkalmazás a felhasználó által megadott város és konyha alapján étel és étteremajánlást biztosít kérdések segítségével. A felhasználó ugyanakkor segíthet az opciók bővítésében azzal, hogy ezeket elküldi az erre megadott felületen az adminisztráció felé. Az adminisztrációnak ugyanakkor biztosított egy olyan felület ahol kezelheti az adatbázis adatait.

A szoftver megvalósítására a Spring és az Angular keretrendszereket választottam. A Spring keretrendszer Java programozási nyelvű alapokra épül, az Angular pedig a JavaScript nyelvre. Ezen keretrendszerek a szoftver fejlesztését nagyban megkönnyítik a különböző, már előre implementált szolgáltatásaikkal. Választásomat nagyban befolyásolta az a tény, hogy ezek a keretrendszerek, technológiák az iparban is keresettek. Adatbázisnak a MongoDB-t választottam, mivel az alkalmazáshoz nincs szükség egy relációs adatbázisra és a funkciók implementálását nagyban megsegíti egy már létező és jól személyre szabható keretrendszer használata.

Az alkalmazás használatához a felhasználóknak és az adminisztrátoroknak nem szükséges komolyabb informatikai tudással rendelkezniük. A felhasználóknak nem szükséges sok inputot megadniuk, mert a választási lehetőségek előre adott igen/nem válaszok, valamint a települések és konyhák neveinek listája is adott. Az adminisztrációs felület annyiban bonyolultabb, hogy az adminisztrátorok az adatbázist kezelik, ezért az inputok nagy része szöveges. A szöveges formátumok miatt lehetnek olyan kérések, amelyek hibás formátumúak.

**2. Felhasznált eszközök, technológiák**

**2.1. Felhasznált eszközök**

**2.1.1. InteliJ Idea**

Az *InteliJ* Idea a JetBrains által fejlesztett Java alapú integrált fejlesztői környezet. Arra fejlesztették ki, hogy nagyban megkönnyítse a fejlesztők feledatát, akik Javában, Kotlinban, Groovyban és más JVM alapú nyelvben fejlesztenek. Nagyon népszerű ez a fejlesztői környezet, hiszen sok segítséget nyújt a programozónak és a számtalan könnyen elérhető biztonságos plugginjaival nagy testreszabhatóságot biztosít.

A választásom azért erre a programra esett, mivel az iparban is nagyon sok helyen használják, könnyű magának a programnak a használata és én magam is már évek óta használom, így jobban ismerem, mint a többi versenytársát.

**2.1.2. WebStorm**

A *WebStorm* szintén egy JetBrains által fejlesztett IDE. A *WebStorm* JavaScript és a hozzá kapcsolódó nyelvekkel való programozás megkönnyítésére lett kifejlesztve. Hasonló az *InteliJ*-hez mind a felhasználó felületében és kezelhetőségében valamint abban, hogy ugyan úgy biztosítja számunkra a nagy testreszabhatóságot.

Azért esett erre a programra a választásom, mert nagyban hasonlít a már említett *InteliJ*-hez és szintén már évek óta használom ezzel megkönnyítve a programom fejlesztését.

**2.2. Felhasznált technológiák**

**2.2.1. MongoDB**

A MongoDB egy nyílt forráskódú NoSQL adatbázis kezelő rendszer, ami egy jó alternatíva a relációs adatbázisokra. Tipikusan nagymennyiségű adatok tárolására és ezek kezelésére szokták használni és dokumentum orientált adattárolásra és kezelésre is képes. Szervezetek még többek közt ad-hoc lekérdezésekre, terhelésmegosztásra és szerver oldali JavaScript futtatásokra is szokták használni.

NoSQL mivoltából a MongoDB nem táblázatszerű architektúrát használ, mint egy relációs adatbázis, hanem gyűjteményekből és dokumentumokból architektúrát. A dokumentumok kulcs-érték párokat tartalmaznak, míg a gyűjtemények (SQL tábláknak felelnek meg) dokumentumok kollekcióit.

A dokumentumok hasonlítanak a JSON-hoz de a MongoDB ennek egy variációját használja, ami a BinaryJSON (BSON). Ennek az előnye az, hogy sokkal több féle adattípust támogat ezzel megkönnyítve a fejlesztők munkáját. A tárolt adatok lehetnek akár másik dokumentumok, tömbök, de akár ezeknek a kombinációja is a felhasználói kézikönyv szerint.

Tehát a MongoDb összegezve:

Miért használják?

* Nagymennyiségű strukturált és strukturáltalan adat tárolása függőleges és vízszintes skálázhatósággal. Keresés mezővel, indexel és kifejezéses lekérdezésekkel.
* Integrált adatok alkalmazásokhoz, beleértve hibrid és többfelhős alkalmazásokat.
* Az adatbázis lehetővé teszi a beágyazott struktúrákat.
* Több szerveren lehet futtatni egyszerre.

Előnyei:

* Séma nélküli, ami lehetővé teszi akármilyen típusú adat tárolását lehetővé téve ezzel az egyszerű skálázhatóságot és nagyobb flexibilitást relációs adatbázisokhoz képest.
* Dokumentum orientált, ami egyszerűbbé teszi a más nyelvekben objektumokká való átfordítást és csökkenti az adatbázis join-ok használatát ezzel csökkentve a költségeket.
* Egyszerű skálázhatóság támogatva az adatbázis szilánkosítását és az adatok zónákba való csoportosítását kulcsokkal.
* Harmadik féltől származó tároló motorok támogatása.
* DBMS-be közvetlen beépített aggregációs lehetőség.

Hátrányai:

* Az automatikus feladatátvételi stratégiájának köszönhetően, ha a mester node nem működik, akkor egy másik node átveszi a feladatát ezzel folytonosságot biztosítva, de ez a folyamat akár egy percet is eltarthat.
* Egy mester node használata miatt az írási mennyiség is limitálva van ezzel potenciális torlódást okozva.
* Nem biztosít, teljes referenciális integritást mivel nem használ idegen kulcsokat.
* Felhasználó hitelesítés nincs alapértelmezetten engedélyezve ezzel rést hagyva potenciális hackereknek bár hozzá lett adva, hogy egy konfigurálatlan adatbázis letilt minden hálózati kapcsolatot.

Továbbá a MongoDB sok programozási nyelvet is támogat, többek között a C, C++, C#, Java, Ruby és a Swift.

**2.2.2. Spring Boot**

A Spring Boot egy Java alapú nyílt forráskódú keretrendszer micro szervizek készítésére. Fejlesztését Pivotal Team végzi, és arra használják, hogy gyorsan gyártásra kész spring alkalmazásokat hozzanak létre.

A Spring Boot egy jó platformot biztosít Java fejlesztők részére, mivel mint már fentebb említve gyorsan önálló gyártási minőségű spring alkalmazást lehet készíteni, amit csak futtatni szükséges. Mindez lehetséges, azért mert elég csak egy minimális konfiguráció kezdésként így nincs szükség elkészíteni egy teljes Spring konfigurációt.

A következő előnyöket biztosít a fejlesztők részére:

* Könnyen érthető és fejleszthető spring alkalmazások.
* Termelékenység növelése.
* Fejlesztési idő csökkentése.

Céljai a Spring Boot-nak:

* Bonyolult XML beállítások elkerülése.
* Egyszerűbb gyártás kész alkalmazás fejlesztése.
* Alkalmazás önálló futtatása és fejlesztési idő csökkentése.
* Egyszerűbb kezdés biztosítása az alkalmazással.

További funkciók:

* Flexibilis Java Bean, XML és adatbázis tranzakció beállítások biztosítása.
* Erős kötegelt feldolgozás és REST végpontok kezelése.
* Minden automatikusan van konfigurálva.
* Annotáció alapú alkalmazás biztosítása.
* Egyszerű függőség kezelés.
* Tartalmaz Beágyazott Servlet Konténert

A Spring Boot automatikusan beállítja az alkalmazásodat a hozzáadott függőségek alapján az **@EnableAutoConfiguration** annotáció használatával. Ugyanakkor egy adatbázist hoz létre a memóriában, ha nincs kapcsolat beállítva az adatbázisomhoz, mindeközben automatikusan megkeresi és beállítja a projektben lévő összes komponenst.

Tehát a Spring Boot egy egyszerűen és gyorsan használható Java keretrendszer, ami nagyban megkönnyíti a programozók életét a széleskörű és mély lefedettségével.

**2.2.3. Angular**

Az Angular egy ingyenes nyílt forráskódú TypeScript alapú webalkalmazás keretrendszer, amit a Google Angular Csapata vezet, magánszemélyek és más vállalatokkal együtt. Az Angular egy Egy Oldal Aplikáció Keretrendszer (SPA) amit gyors webalkalmazások készítésére használnak. Az SPA koncsepciója az, hogy csak egyszer kelljen, az oldalt betölteni utána csak adatokat kelljen kérnünk a szervertől, ami gyorsabbá teszi a webalkalmazásunkat.

A keretrendszert HTML és TypeScript segítségével tudjuk használni és különböző funkciókat megvalósítani. Maga a rendszer is TypeScript-ben íródott, ami implementál különböző funkcionalitásokat, amit mi is importálhatunk a saját alkalmazásunkba.

**2.2.4. TypeScript**

A TypeScript egy magas szintű programozási nyelv, amit a Microsoft fejleszt és tart karban. A programozási nyelv a JavaScript egy alfaja mivel hozzáad egy opcionális szigorú típusitást. Arra lett tervezve, hogy nagyméretű alkalmazások fejlesztésére használják és mivel a JavaScript-nek csak egy alfaja ezért át lehet fordítani JavaScript kóddá és ugyan ezt meg lehet csinálni fordítva is.

**2.2.4. Bootstrap**

A Bootstrap egy ingyenes nyílt forráskódú CSS keretrendszer, amit a reszponzív front-end webfejlesztésre fejlesztettek ki. HTML, CSS és JavaScript könyvtárakat tartalmaz, amik célja az informatív weboldalak készítésének a megkönnyítése.

Az elsődleges célja a Bootstrap-nek az hogy egy már előre definiált kinézeteket tudjunk adni a weboldalunkon megtalálható elemeknek és ezek a kinézetek reagálni tudjanak a megjelenítő eszközök tulajdonságaira, főként a méretre. Fő célja ezzel az, hogy minden megjelenítő eszközön ugyan úgy nézzen ki a weboldalunk és ennek a megvalósítását könnyebbé tegye számunkra. Ugyanakkor lehetővé teszi számunkra, hogy a már definiált elemeket szeméjre szabjuk a saját szükségleteinkre.

JavaScript komponenseket is tartalmaz, amikhez nincs szükség külső könyvtárakra. Ezek főként a dinamikus, interakcióba léphető elemeknél lehet megtalálni, mint például egy legördülő mező vagy folyamatjelző sáv. Néhány komponense a keretrendszernek ugyanekkor ki bővítheti a már létező interfész elemet, mint például a bemeneti mezők automatikus kiegészítése, de a legfőbb feladata és komponensei a weboldal elrendezéséért felelősek.

**3. Feladat leírása**

**3.1. Téma**

Mint már azt az előbbiekben is említettem már az alkalmazásom fő témája az, hogy megkönnyítsem az emberek döntéshozatalát abban, hogy egy-egy étkezés során mit egyenek és azt a megadott városukban hol kapható.

Ezt a témát főként azért választottam mivel jómagam is, mint sokan mások, gyakran szenvedek attól, hogy nem tudom, mit szeretnék pontosan enni csak azt, hogy milyen ízvilágú ételt kívánok abban a pillanatban. Ugyanakkor gyakran azt se tudom, hogy az adott városban ahol éppen tartózkodom, milyen éttermek vannak, és azok mit szolgálnak fel. Ezért döntöttem úgy hogy megpróbálok egy olyan alkalmazást készíteni, ami valamilyen formában a fenti problémáimat meg tudja oldani.

**3.2. Hasonló alkalmazások**

Bár már léteznek, hasonló alkalmazások a legtöbbjük inkább fókuszál az éttermekre és az ételrendelésre mintsem arra, hogy segítsen eldönteni mit, egyél. Amelyik alkalmazás pedig segít ételt választani azok csak egy, a felhasználó által megadott listából választanak ki egyet, ami nem oldja meg azt a problémát, ha teljesen nem tudjuk mit együnk vagy esetleg valami új ételt is megkóstolnánk, amiről nem tudunk.

Jelenleg (dolgozat írásának idején) legjobb tudomásom szerint nem is létezik olyan alkalmazás, ami kifejezetten ételt ajánl fel a felhasználónak, hanem éttermeket sorol fel a megadott szűrési feltételekre. Amelyik alkalmazások hasonlóak még azok az ételrendelő alkalmazások, mint például a Wolt és a Foodpanda viszont ezek az alkalmazások csak az éttermekre szűrnek és azok a szűrési feltételek is eléggé tágak ezzel megnehezítve a dolgunkat. Ugyanakkor ezeknél az alkalmazásoknál néha eléggé pontatlanul vannak megadva, néha teljesen el is hagyva azokat, hogy egy-egy étterem milyen témájú és mit szolgál fel, ami megnehezíti és pontatlanná teszi az éttermek szűrését.

Ezért az én alkalmazásom az előbbiekben említetteket hibákon szeretne javítani és a ki nem használt lehetőségeket kihasználni ezzel egy a piacon lévő lyukat betömni. Ugyanakkor az alkalmazásom nem próbálná elvenni a már piacon lévők létjogosultságát, hanem azokat kiegészíteni.

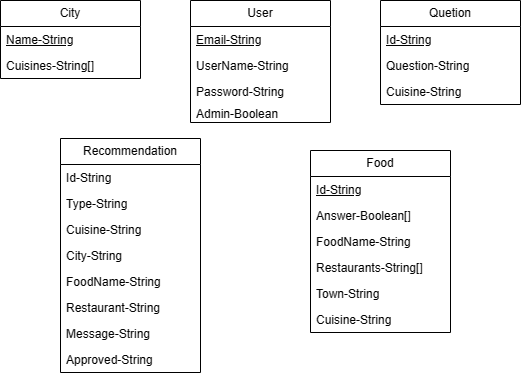
**4. Fejlesztői leírás**

**4.1. Back end**

**4.1.1. Adatbázis**

Az adatbázisom tervezésénél a fő szempontom az volt, hogy a lehető legkevesebb táblából álljon miközben ezek a táblák felépítése a lehető legátláthatóbb és legtöbb adatot tárolják, amiket együtt kezelne a rendszer. Ezek a kritériumok miatt és az alkalmazás mivoltijából adódóan sikerült egy olyan adatbázist létrehoznom amiben nincsennek relációk, bár van olyan helyzet az alkalmazás futása során, amikor egy-egy adat függhet a másiktól ezek annyira riták és specifikusak, hogy egyszerűbb ezeket esetspecifikusan a probléma felmerülésénél megoldani, mint az egész adatbázist e-köré tervezni és fölöslegesen bonyolítani ezzel azt.

Ezért a már korábban említett MongoDB szolgáltatásait vettem igénybe az adatbázisomhoz, hiszen ez a rendszer biztosít egy ingyenes és könnyen használható NoSQL adatbázist, ami az én céljaimnak tökéletesen megfelel. Ugyanakkor támogat több külső keretrendszert is, amik nagyban megkönnyítik az adatbázissal való kommunikációt a fejlesztő részére.



*1. ábra – Adatbázis táblák és mezőik*

Az ábrán (1. ábra) látható az adatbázisban megtalálható összes tábla, azok mezői és típusai. Több táblában is látszik, hogy tartalmaznak tömböket és egyes táblákban van olyan mező, amiknek van relációja az adatbázis ennek ellenére sem relációs hanem NoSQL. Bár meglehetett volna oldani az adatbázist relációskén a sok tömb miatt az egyes táblákban sok kapcsolótábla jönne létre, ami nagyban lassította volna az adatbázis működését hisz abból kifolyólag, hogy egy-egy konyhának akár több száz különböző étele is lehet és több városban is lehetnek jelen éttermek, amik ezeket felszolgálják, elkerülhetetlen a nagymennyiségű adat.

Ezért is döntöttem úgy, mivel már az előbbiekben is említettem, hogy ebben a fajta adatbázisban, a nagymennyiségű adatban a keresés gyorsabb és megéri ezt használni és kis kellemetlenségek, amit okozhat ebből kifolyólag.

**4.4.2.**