Tartalom

[1.Bevezetés 2](#_Toc100037696)

[2.Fejlesztői eszközök 3](#_Toc100037697)

[2.1.Visual Studio Code 3](#_Toc100037698)

[2.2. Node.js 3](#_Toc100037699)

[2.3. MongoDB 3](#_Toc100037700)

[3. Fejlesztői környezet 4](#_Toc100037701)

[3.1. JavaScript 4](#_Toc100037702)

[3.2. React 4](#_Toc100037703)

[3.Frontend 6](#_Toc100037704)

# 1.Bevezetés

Szakdolgozat témájának egy online ruha rendelős web program tervezését és kivitelezését választottuk. Napjainkban sokan választják már inkább az online vásárlást és ezt a dolgot szerettük volna minél felhasználóbaráttá, könnyen kezelhetővé és átláthatóbbá tenni, hogy azok számára is egyszerű legyen a kezelése, aki még nem csinált ilyet és nem jártas benne. A programunk tartalmaz egy regisztrációt, ami után az adatokat eltároljuk, és későbbiekben segít a felhasználónak, hogy rendelés leadásánál már ne keljen még egyszer megadni a személyes adatait.

# 2.Fejlesztői eszközök

## 2.1.Visual Studio Code

A program kódot főként a visual studo codeban készítettük, de emelet pár másik dolog is kellet hozzá. A Visual Studio Code (rövidítve: VSCode vagy VS Code) egy ingyenes, nyílt forráskódú kódszerkesztő, melyet a Microsoft fejleszt Windows, Linux és OS X operációs rendszerekhez. Támogatja a hibakeresőket, valamint beépített Git támogatással rendelkezik, továbbá képes az intelligens kódkezelésre (intelligent code completion) az IntelliSense segítségével. Ezen felül testre szabható, így a felhasználók megváltoztathatják a kinézetet (témát), megváltoztathatják a szerkesztő gyorsbillentyű-kiosztását, az alapértelmezett beállításokat és még sok egyebet. A Visual Studio Code az Electron nevű keretrendszeren alapszik, amellyel asztali környezetben futtatható Node.js alkalmazások fejleszthetőek. Ugyanakkor a Visual Studio Code nem az Atom forkja, hanem a Visual Studio Online szerkesztőn alapszik (fejlesztési neve: "Monaco").

## 2.2. Node.js

A backend megvalósításához szükségünk volt a Node.js rendszerre. A Node.js egy szoftverrendszer, melyet skálázható internetes alkalmazások, mégpedig webszerverek készítésére hoztak létre.  A programok JavaScriptben írhatók, eseményalapú, aszinkron I/O-val a túlterhelés minimalizálására és a skálázhatóság maximalizálására. A Node.js a Google-féle V8 JavaScript-motorból, a libUV-ből és számos beépített könyvtárból áll. Magában a JavaScript motor képtelen arra, hogy szervert lehessen futtatni rajta, mivel neki az az elsődleges feladata a komponensek kezelése és kérések kiküldése, erre nyújt megoldás a Node.js, amelynek, elsődlegesen az „express” harmadik féltől származó könyvtárra volt szükségünk, mivel ez képes kényszeríteni a Javascript motort az abnormális viselkedésre. Az express ugyanis képes rávenni a motort, hogy portokat tudjon figyelni és esedleges kérések esetén válaszolni is tudjon. Egy másik fontos könyvtár a „nodemon”, erre azért volt szükségünk mivel a .js kiterjesztésű fájlokat nem tudjuk futtatni a böngészőben megtalálható JS motor kihagyásával, amire pedig szükségünk lenne, bár erre egyéb megoldások is léteznek, de erre esett a választásunk, mivel véleményünk szerint ez a legelegánsabb. Tehát a nodemon képes futtatni a script fájlunkat konzolos környezetben.

## 2.3. MongoDB

Az adatbázishoz MongoDB-t használtunk mivel ezt találtuk a legmegfelelőbbnek a projektünkhez. A MongoDB nyílt forráskódú dokumentumorientált adatbázis szoftver, amelyet a MongoDB inc. (korábbi nevén *10gen*) fejleszt. A NoSQL adatbázisszerverek közé tartozik. A dokumentumokat JSON-szerű formátumban tárolja (BSON). A legnépszerűbb NoSQL adatbázis szoftver. A MongoDB fejlesztését 2007-ben kezdték a 10gen-nél, amikor a cég egy platform szolgáltatás fejlesztésén dolgozott. 2009-ben a szoftvert. Az 1.4-es verzió 2010 márciusi kiadásával a fejlesztő csapat éles üzemre késznek tartja a terméket.

# 3. Fejlesztői környezet

## 3.1. JavaScript

A JavaScript programozási nyelv egy objektumorientált, prototípus-alapú szkriptnyelv, amelyet weboldalakon elterjedten használnak. Eredetileg *Brendan Eich*, a Netscape Communications mérnöke fejlesztette ki; neve először *Mocha*, majd *LiveScript* volt, később „JavaScript” nevet kapott, és szintaxisa közelebb került a Sun Microsystems Java programozási nyelvéhez. A JavaScriptet először 1997–99 között szabványosította az ECMA „ECMAScript” néven. A jelenleg is érvényes szabvány az *ECMA-262 Edition 3* (1999. december), ami a *JavaScript 1.5*-nek felel meg. Ez a szabvány egyben ISO szabvány is. A Microsoft 1995-ben kifejlesztette az Internet Explorert, ami a Netscape-el való böngészőháborúhoz vezetett. A Microsoft a Netscape Navigator JavaScript-feldolgozójának forráskódja segítségével létrehozta a sajátját, a JScriptet. A JScript először 1996-ban jelent meg a CSS kezdeti támogatása és a HTML néhány kiegészítése mellett. E megvalósítások merőben különböztek a Navigatorban alkalmazottaktól, ami megnehezítette a fejlesztőknek, hogy a weblapjaik mindkét webböngészőben jól működjenek, ami a „Netscape-en működik legjobban” és „Internet Exploreren működik legjobban” széles körű használatához vezettek sok éven át.

## 3.2. React

A React kód komponenseknek nevezett entitásokból áll . Ezek az összetevők újra felhasználhatók, és az SRC mappában kell őket létrehozni a Pascal Case névváltoztatást követően Az összetevők a DOM egy adott eleméhez renderelhetők a React DOM könyvtár használatával. Egy komponens renderelésekor az értékeket a komponensek között átadhatjuk a " props "-on keresztül.

# 3.Frontend

A frontend része Reactban készült el és próbáltuk minél egyszerűbbé tenni mivel szerintünk az egyszerű néha jobb alapon és szerintünk ez sikerült is.

A legtöbb dolgot kártyákkal valósítottuk meg. Azért ezt választottuk mivel ez tűnt a legjobbnak és egyszerűnek a termékek és az ötletünk megvalósításához megvalósításának módjára. Az oldal legjobb megjelenítés érdekében bootstrapet használtunk ez által telefonon és egyéb eszközökön is megfelelő a látvány, kivitelezés. A következő oldalakon látni lehet majd az oldalunk felépítését, kódolását.