A megvalósítandó szoftver követelmény leírása

**Áttekintés**

Egy komplex rendszer kifejlesztése hosszú időt, tervezést és odafigyelést igényel, amit befolyásol a projekt mérete és az azon dolgozó csapatok száma. Minél összetettebb a rendszer, minél több fejlesztőcsapat dolgozik a projekten, annál komolyabb munka a csapatok lokális és globális menedzselése. A szoftver minősége és a határidők betartása függ az adott csapat készségétől, a tervezéstől és az előző fejlesztési ciklusok alapján kialakult tanulságoktól, valamint a csapattagok motiváltságától. A szakdolgozat témája egy olyan applikáció elkészítése, amelyben lehetőség nyílik egy agilis módszertan alapján dolgozó fejlesztőcsapat menedzselésére és támogatására, beleértve a munkafolyamatok megtervezésének gördülékennyé tételét, az adott fejlesztési ciklus, azaz ‘sprint’ kiértékelését csapat szinten, ezek visszakereshetőségét és feladatkörökre való osztását, a csapatmorál mérését és metrikák generálását a még hatékonyabb tervezés érdekében, valamint kompetencia mátrix előállítását a csapat tudását illetően.

Cél egy olyan alkalmazás lefejlesztése és prezentálása, ami egy valódi agilis csapatban jelentkező problémák megoldását teszi könnyebbé, gyorsabbá, és átláthatóbbá, a lehető legnagyobb transzparencia mellett. A Célközönség egy agilis módszertan alapján fejlesztő scrum csapat, ami a fejlesztőkből, egy scrum masterből, és egy product owneből áll.

Funkcionalitás szempontjából a legfontosabb, hogy átláthatóak és könnyen kezelhetőek legyenek, amiket a módszertan megkövetel, valamint az agilitáshoz megfelelően maximális rugalmasságot tegyen lehetővé. (A fent említett Scrum módszertanok később kerülnek kifejtésre). Az elkészített szoftver egy a böngészőből elérhető webes alkalmazás.

**Rendszer Funkciói**

Az elkészített web alkalmazás egy a legmodernebb szoftvertechnológiák által ötvözött úgynevezett web-stack. A web-stack szoftverek és technológiák kollekciója, amik egy komplex alkalmazás lefejlesztéséhez szükségesek. A Fő alkotó elemük egy Frontend service egy Backend service egy adatbázis motor, ami általában SQL adatbázis, tehát nem relációs adatbázis, ezek általában JSON alapúak. Ezen szakdolgozat témáját alkotó alkalmazás egy a manapság legmodernebb technológiák felhasználásával alkotott full-stack web alapú szoftver, ami a (MySQL, Express.js, React, Node.js)-ből áll.

* MySQL
  + A MySQ egy [többfelhasználós](https://hu.wikipedia.org/w/index.php?title=T%C3%B6bbfelhaszn%C3%A1l%C3%B3s&action=edit&redlink=1), [többszálú](https://hu.wikipedia.org/w/index.php?title=T%C3%B6bbsz%C3%A1l%C3%BAs%C3%A1g&action=edit&redlink=1), [SQL](https://hu.wikipedia.org/wiki/SQL)-alapú [relációs adatbázis-kezelő](https://hu.wikipedia.org/wiki/Rel%C3%A1ci%C3%B3s_adatb%C3%A1zis-kezel%C5%91) [szerver](https://hu.wikipedia.org/wiki/Szerver).
  + <https://hu.wikipedia.org/wiki/MySQL>
* Express.js
  + Az Express egy gyors, rugalmas, de mégis kis web framework, amiben könnyedén lehet robosztus web alkalmazásokat írni Node.js alapokon.
  + <https://expressjs.com/>
* React
  + A React egy komponens alapú javascript frontend könyvtár, felhasználói interfészek megjelenítésére. <https://reactjs.org/>
* Node.js:
  + A **Node.js** egy szoftverrendszer, melyet [skálázható](https://hu.wikipedia.org/wiki/Sk%C3%A1l%C3%A1zhat%C3%B3s%C3%A1g) internetes alkalmazások, mégpedig [webszerverek](https://hu.wikipedia.org/wiki/Webszerver) készítésére hoztak létre. A programok [JavaScript](https://hu.wikipedia.org/wiki/JavaScript)-ben írhatók, [eseményalapú](https://hu.wikipedia.org/wiki/Esem%C3%A9nyalap%C3%BA_szoftverarchitekt%C3%BAra), aszinkron I/O-val a túlterhelés minimalizálására és a [skálázhatóság](https://hu.wikipedia.org/wiki/Sk%C3%A1l%C3%A1zhat%C3%B3s%C3%A1g) maximalizálására.
  + <https://hu.wikipedia.org/wiki/Node.js>

Egyéb felhasznált modulok / fejlesztőeszközök / kód könyvtárak:

Ezeket a fejlesztés során külön kell installálni CLI (command line interface) segítségével, az *npm install* paranccsal. Használva a cli adta *--save* kapcsolót lehetőség van a függőségeket nem globálisan, hanem lokálisan az adott projekt mappájába telepíteni, ekkor a projekt *package.json* fájljába kerül elmentésre a *dependencies* kulcsszó alatt a függőség neve, valamint a projekt mappájában létrejövő *node\_modules* mappába történik az installálás. Az így létrejövő projektet, ha szeretnénk egy másikeszközön használni, akkor lehetőségünk nyílik egy paranccsal feltelepíteni az összes függőséget, feltéve, ha mindegyik szerepel a *package.json* fájlban. Ez esetben a megfelelő helyen állva a projekt mappában, kiadva az *npm install* parancsot az npm (node package manager) felolvassa a *package.json dependencies* alatt lévő listát, és feltelepíti mindegyiket.

Backend:

* cors
  + Middleware kiegészítő Express.js-hez, ami lehetővé teszi a CORS (Cross Origin Resource Sharing) módszert, ezáltal a modulok és rétegek közötti adatfolyamot.
  + <https://www.npmjs.com/package/cors>
* passport
  + Express.js kompatibilis middleware felhasználói autentikációra. Különböző előre definiált *Strategy* található meg benne, amikkel elérhetőek a legnépszerűbb közösségi belépési módszerek, mint a Facebook, Google, vagy az Instagram. JWT ( Json Web Token) alapú, ami azt jelenti, hogy ha a belépési kérés *request* sikeres, akkor a rendszer amit használunk a válaszban egy JWT-t küld *response* benne a felhasználó adataival, és a belépésre való jogosultsággal.
  + <https://www.npmjs.com/package/passport>
* chalk
  + Segítségével tudunk különböző színekkel üzeneteket kiírni a terminál képernyőjére, ezáltal elkülöníteni az eltérő üzeneteket.
  + <https://www.npmjs.com/package/chalk>
* bodyParser
  + Átalakítja a szerver felé beérkező kéréseket.
  + <https://www.npmjs.com/package/body-parser>
* Sequelize
  + Többirányú *Promise* alapú ORM (Object Relational Management) kezelő a legelterjedtebb SQL (Structured Query Language) verziókhoz, mint a Posgres, MySQL, MariDB.
* concurrently
  + Segítségével több CLI parancsot tudunk futtatni egyszerre, egyidőben. A szakdolgozat alapján elkészült alkalmazás esetén egyszerre egy parancs segítségével indul mind a server és a kliens alkalmazás is.
  + <https://www.npmjs.com/package/concurrently>
* mysql2
  + Gyorsaságot előnyben részesítő mysql kliens.
  + <https://www.npmjs.com/package/mysql2>
* nodemon
  + Minden a szerver fájlt ért változtatás után újra indítja a szervert az új frissítésekkel.
  + <https://www.npmjs.com/package/nodemon>

Frontend:

* Material-ui
  + Google által fejlesztett React komponensek gyűjteménye.
  + <https://www.npmjs.com/package/@material-ui/core>
* lodash
  + Gyors és könnyű adatkezelés és adat manipuláció.
  + <https://lodash.com/>
* react-router
  + Útvonal meghatározás az oldalak, és komponensek között frontend oldalon.
  + <https://www.npmjs.com/package/react-router>