



# Shopping List

Dokumentace k ročníkové práci

**Autor:** Martin Kubjak

**Třída:** 3ITB

**Vedoucí práce:** Bc. Jakub Pokorný

2021/2022

## Prohlášení

---

Prohlašuji, že jsem ročníkovou práci na téma „Shopping List“ vypracoval samostatně a s použitím uvedené literatury a pramenů.

V (název obce, kde podepisuji) dne .....

.....

## Poděkování

---

Chtěl bych poděkovat Bc. Jakub Pokorný za vedení mé ročníkové práce, cenné rady a odborný dohled.

# Anotace

---

Tento dokument je dokumentace k mé práci. V úvodu práce je popis co je v této práci. V technologiích je vypsáno jaké technologie jsem použil. V praktické části jsem vypsál úplně všechny funkce a jejich kód.

## 1 Klíčová slova

---

React, nodejs, javascript, express

# Obsah

---

Anotace .....	4
1 Klíčová slova .....	4
Obsah.....	5
Úvod .....	6
2 Technologie .....	7
3 Praktická část .....	8
3.1 Návrhy.....	8
3.1.1 Databáze.....	8
3.2 Produktizace .....	9
3.2.1 Login Systém.....	9
3.2.2 Register Systém .....	9
3.2.3 Změna údajů na účtu.....	10
3.2.4 Managment listů .....	12
3.2.5 Upravení položky .....	13
3.2.6 Změnění informací o položky .....	14
3.2.7 Mazání layoutu .....	15
3.2.8 Statistiky nákupů .....	15
3.2.9 Přidání do rodiny .....	16
3.2.10 Přidání ceny nákupu a označení jako hotový .....	16
3.3 Popis pro uživatele .....	16
Použitá literatura .....	17
Seznam obrázků.....	17
Obsah média.....	18

# Úvod

---

Můj cíl pro tuto ročníkovou práci byl, abych usnadnil tvorbu seznamů a orientaci zákazníků při nakupování v neznámých obchodech s potravinami. V druhém pololetí plánuji přidat řazení seznamu podle toho, jak je to v určitém obchodě, aby se uživatel nemusel vracet. Aplikace umí vytvářet seznamy, přidávat položky, vymazávat položky, upravovat položky, změnit stav z koupeno na nekoupeno, změnit si jméno, heslo, email a vymazat si účet.

## 2 Technologie

---

Node.js – „Node.js je softwarový systém navržený pro psaní vysoce škálovatelných internetových aplikací, především webových serverů.“ (1)

React - „React je Javascriptová knihovna pro tvorbu uživatelského rozhraní.“ (2)

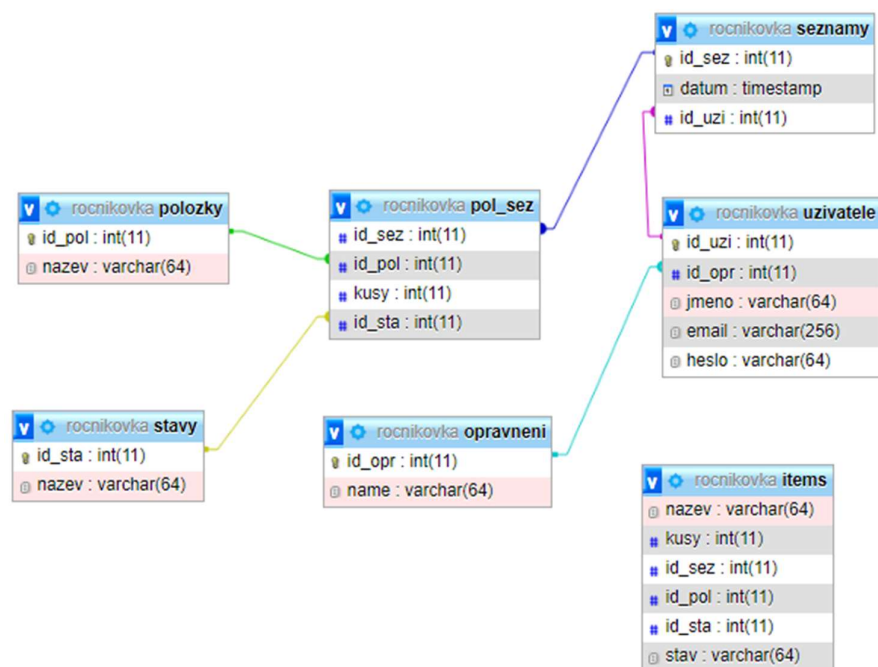
Express - „Express je back-end webový aplikační rámec pro Node.js.“ (3)

MySQL – „MySQL je otevřený systém řízení báze dat uplatňující relační databázový model.“ (4)

## 3 Praktická část

### 3.1 Návrhy

#### 3.1.1 Databáze



Obrázek 1

V tabulce polozky ukládám názvy položek, které následně používám ve spojovací tabulce pol\_sez, ve které jsou všechny informace o všech seznamech a stavech (koupené, nekoupené). V tabulce uzivatele jsou uloženy informace o uživateli, jméno, heslo (které je zahashované pomocí SHA1), email a id\_opr. V tabulce opraveni je název role např. Admin, normal user... Potom tu je view items do kterého jsem dal všechny informace o každém itemu a potom je používám na to, aby se mi lépe mazaly itemy, zobrazovaly a měnily informace o nich.



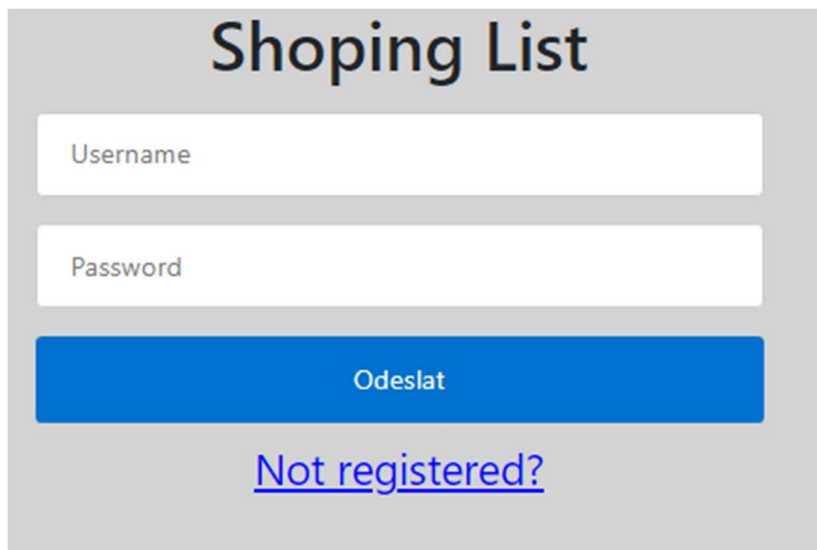
## 3.2 Produktizace

### 3.2.1 Login Systém

```
function login(username, password, res) {
  con.query("SELECT * FROM uzivatele", function (err, result, fields) {
    if (err) throw err; //pokud je error pri pripojovani k db tak hodi error
    for (let i = 0; i < result.length; i++) {
      if (username == result[i].jmeno && passwordHash.verify(password, result[i].heslo)) {
        return res.send({login: true, id: `${result[i].id_uzi}`}); //kontroluje pokud je spravne heslo i jmeno
      }
    }
    res.send({login: false})
  });
}
```

Obrázek 2

Do této funkce na serveru se přijmou parametry username a password, které zadal uživatel do login formu.



Obrázek 3

Ve funkci se poté „selectne“ každý uživatel z databáze a projede se pokud tam je jméno a heslo a pokud to je správná kombinace. Výstup je json objekt s 2 proměnnými login a id pokud je heslo i jméno správně odešle se login jako true a id jako id uživatele pokud ne odešle se login false.

### 3.2.2 Register Systém

```
function register(username, password, email, res) {
  var hashedPassword;

  con.query("SELECT jmeno FROM uzivatele", function (err, result, fields) {
    //kontroluje pokud jmeno v db
    if (err) throw err; //pokud je error pri pripojovani k db tak hodi error
    for (let i = 0; i < result.length; i++) {
      if (username == result[i].username) {
        return res.send({message: "Username not available choose another one"}); //kontroluje pokud jmeno není v db
      }
    }
    hashedPassword = passwordHash.generate(password); //převádění hesla
    var sql = "INSERT INTO uzivatele (jmeno, heslo, email, id_uzi) VALUES ('${username}', '${hashedPassword}', '${email}', 2)"; //vloží jméno a heslo
    con.query(sql, function (err, result) {
      if (err) throw err; //pokud je error pri pripojovani k db tak hodi error
    });
    res.send({message: "Created an account", "created": true});
  });
}
```

Obrázek 4

V této funkci na serveru se přijmou parametry username, password a email z register formu od uživatele

# Shoping List

Obrázek 5

Na frontendu se nejdříve zkontroluje pokud obě hesla jsou stejná a potom se POSTne na server. Tam se zkontroluje pokud jméno již existuje a pokud ne tak se zahashuje heslo a přidá se všechno do databáze.

### 3.2.3 Změna údajů na účtu

#### 3.2.3.1 Změna jména

```
const changeUsername = (id_uzi, username) => {
  sql = `UPDATE uzivatele SET jmeno="${username}" WHERE id_uzi = ${id_uzi}`
  con.query(sql, (err, result) => {
    if(err) throw err
  })
}
```

Obrázek 6

## Change Username:

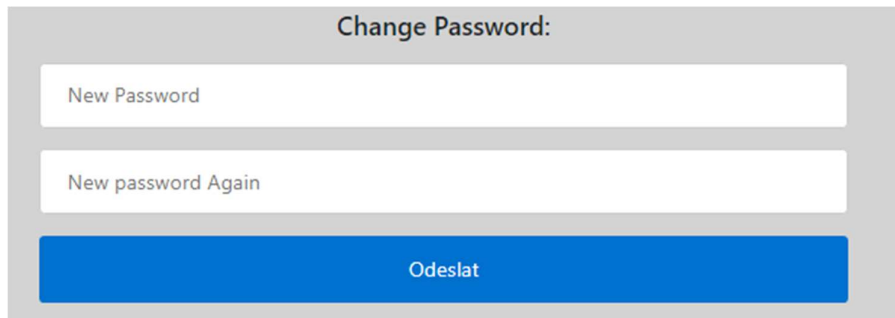
Obrázek 7

Zde si uživatel může změnit přihlašovací jméno.

### 3.2.3.2 Změna hesla

```
const changePassword = (id_uzi, password) => {  
  const hashedPassword = passwordHash.generate(password)  
  sql = `UPDATE uzivatele SET heslo='${hashedPassword}' WHERE id_uzi = ${id_uzi}`  
  con.query(sql, (err, result) => {  
    if(err) throw err  
  })  
}
```

Obrázek 8



A web form titled "Change Password:". It contains two text input fields: "New Password" and "New password Again". Below these fields is a blue button labeled "Odeslat" (Submit).

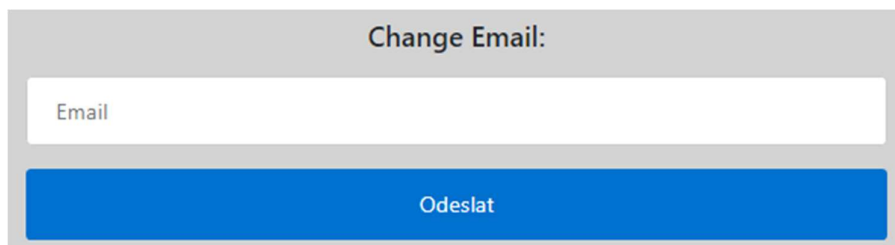
Obrázek 9

Zde si uživatel může změnit heslo.

### 3.2.3.3 Změna emailu

```
const changeEmail = (id_uzi, email) => {  
  sql = `UPDATE uzivatele SET email='${email}' WHERE id_uzi = ${id_uzi}`  
  con.query(sql, (err, result) => {  
    if(err) throw err  
  })  
}
```

Obrázek 10



A web form titled "Change Email:". It contains a single text input field labeled "Email". Below this field is a blue button labeled "Odeslat" (Submit).

Obrázek 11

Zde si uživatel může změnit email.

### 3.2.3.4 Vymazání účtu

```
const deleteAccount = async (id_uzi) => {
  sql = `SELECT * FROM seznamy WHERE id_uzi = ${id_uzi};`
  const seznamy = await new Promise((resolve, reject) => {
    con.query(
      sql,
      (err, result) => {
        return err ? reject(err) : resolve(result);
      }
    );
  });

  seznamy.map((seznam) => {
    list.deleteList(seznam.id_sez)
  })

  sql = `DELETE FROM uzivatele WHERE id_uzi = ${id_uzi};`

  con.query(sql, (err, result) => {
    if(err) throw err
  })
}
```

Obrázek 12

Zde se vymaže celý účet i se všemi jeho seznamy. Nejdříve se se „selectnou“ všechny seznamy toho uživatele a zadají se do json objektu. Po zmapování se každý seznam vymaže. Po smazání seznamů se odstraní i účet.

### 3.2.4 Managment listů

#### 3.2.4.1 Vytvoření listu

```
function createList(id_uzi) {
  sql = `INSERT INTO seznamy(id_uzi) VALUES ("${id_uzi}")`
  con.query(sql, function (err, result) {
    if(err) throw err;
  })
}
```

Obrázek 13

Po tom co klikne uživatel na tlačítko „Create List“ se zavolá funkce, ve které je POST na server s id\_uzi, které je uloženo ve frontendu v useState.

### 3.2.4.2 Přidání položky do listu

```
function addItem(item, id_sta, id_sez, kusy) {  
  sql = `INSERT INTO polozky (nazev) VALUES ("${item})"`  
  con.query(sql, function (err, result) {  
    if(err) throw err  
  })  
  con.query(`SELECT id_pol FROM polozky WHERE nazev = "${item}"`, function (err, result) {  
    if(err) throw err  
    sql = `INSERT INTO pol_sez(id_sez, id_pol, kusy, id_sta) VALUES (${id_sez}, ${result[result.length-1].id_pol}, ${kusy}, ${id_sta})`  
    con.query(sql, function (err, result) {  
      if(err) throw err  
    })  
  })  
}
```

Obrázek 14

Po zadání dat do formu, který přidává položky, dochází k zavolání POSTu na server s názvem položky, počtem kusů, id seznamu a id položky.

### 3.2.4.3 Vymazání listu

```
const deleteList = (id_sez) => {  
  sql = `DELETE FROM pol_sez WHERE pol_sez.id_sez = ${id_sez};`  
  con.query(sql, (err, result) => {  
    if(err) throw err  
  })  
  
  sql = `DELETE FROM seznamy WHERE id_sez = ${id_sez};`  
  con.query(sql, (err, result) => {  
    if(err) throw err;  
  })  
}
```

Obrázek 15

Po kliknutí na tlačítko koš se zavolá POST s id listu a vymaže se list i se všemi položkami.

### 3.2.5 Upravení položky

#### 3.2.5.1 Vymazání položky

```
const deleteItem = (id_pol) => {  
  sql = `DELETE FROM polozky WHERE polozky.id_pol = ${id_pol};`  
  
  con.query(sql, function (err, result) {  
    if (err) throw err;  
  })  
  
  sql = `DELETE FROM pol_sez WHERE pol_sez.id_pol = ${id_pol};`  
  con.query(sql, function (err, result) {  
    if (err) throw err;  
  })  
}
```

Obrázek 16

Po kliknutí na tlačítko koš se zavolá POST s id položky a vymaže se z tabulky položky i spojovací tabulky pol\_sez.

### 3.2.5.2 Změnění stavu itemu

```
const changeState = (id_sta, id_pol) => {  
  sql = `UPDATE pol_sez SET id_sta='${id_sta}' WHERE id_pol = ${id_pol}`  
  con.query(sql, (err, result) => {  
    if (err) throw err  
  })  
}
```

Obrázek 17

Po kliknutí na křížek/fajfku se zavolá POST s id položky a id stavu a změní se to.

### 3.2.6 Změnění informací o položce

```
const changeItem = (id_pol, nazev, kusy) => {  
  sql = `UPDATE pol_sez SET kusy=${kusy} where id_pol = ${id_pol}`  
  con.query(sql, (err, result) => {  
    if (err) throw err  
  })  
  sql = `UPDATE polozky SET nazev='${nazev}' where id_pol = ${id_pol}`  
  con.query(sql, (err, result) => {  
    if (err) throw err  
  })  
}
```

Obrázek 18

Po odeslání formu se zavolá POST s id položky, novým názvem a novým počtem kusů a dojde ke změně.

#### 3.2.6.1 Vytvoření layoutu

Name of da shop

City of da shop

Send It

-	alko	děti	elektronika	koupelna	maso
mléčné výrobky	nádobí	nealko	oblečení	ovoce a zelenina	pečivo
sladkosti	uzeniny	zahrada	zamražené		

Obrázek 19

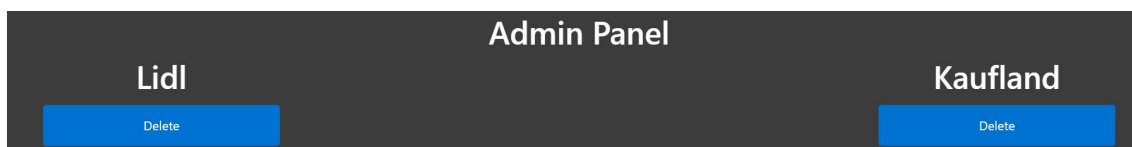
Pro vytvoření nového layoutu a obchodu je tento formulář. Po kliknutí na typ produktu se přidá do prázdného rámečku a запиše se do pole. Aby se mohl form odeslat musíme přidat minimálně 8 typů. Po odeslání se POSTne na server a tam se každá položka propíše do DB každá položka na jeden řádek.

id_mark	pozice	id_typ
0	0	1
0	1	3
0	2	12
0	3	13
0	4	9
0	5	8
0	6	5
0	7	14
0	8	16
0	9	11
0	10	4
0	11	7
0	12	2
0	13	10
0	14	6
0	15	15

Obrázek 20

### 3.2.7 Mazání layoutu

Pro vymazání layoutu se vypíše všechny názvy obchodů a pod nimi tlačítko Delete po jeho zmáčknutí se POSTne na server a tam se vymaže seřazení a celý obchod.



Obrázek 21

### 3.2.8 Statistiky nákupů

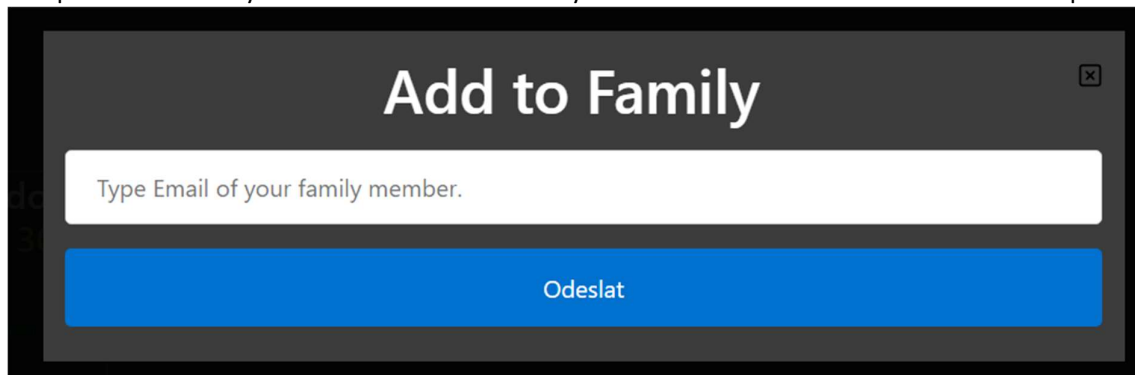
Pro vypočítání statistik je potřeba dokončit nákup a přidat k němu cenu.



Obrázek 22

### 3.2.9 Přidání do rodiny

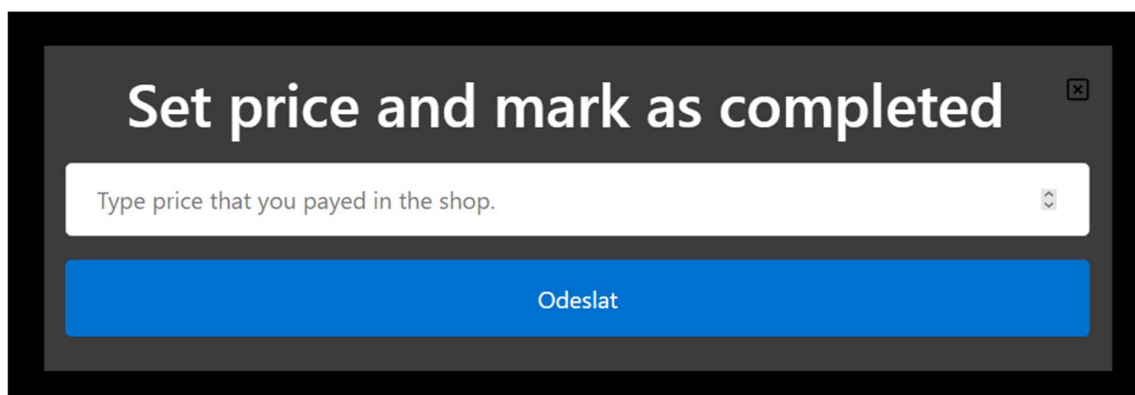
Pro přidání do rodiny klikneme na tlačítko rodiny a tam zadáme email toho koho chceme přidat.

A screenshot of a web form titled "Add to Family" in a dark grey modal window. The title is in large white font. In the top right corner of the modal is a small square button with a white 'X' icon. Below the title is a white text input field with the placeholder text "Type Email of your family member." in grey. Below the input field is a wide blue button with the white text "Odeslat".

Obrázek 23

### 3.2.10 Přidání ceny nákupu a označení jako hotový

Klikneme na tlačítko Show a 2 pod ním je tlačítko Mark as Completed.

A screenshot of a web form titled "Set price and mark as completed" in a dark grey modal window. The title is in large white font. In the top right corner of the modal is a small square button with a white 'X' icon. Below the title is a white text input field with the placeholder text "Type price that you payed in the shop." in grey. To the right of the input field is a small upward and downward arrow icon. Below the input field is a wide blue button with the white text "Odeslat".

Obrázek 24

## 3.3 Popis pro uživatele

---

Po spuštění stránky se uživatel přihlásí pomocí jména a hesla.

Uživatel si může vytvořit list, do kterého lze přidat položky. Po kliknutí na tlačítko „Profil“, si uživatel může změnit jméno, heslo a email, případně vymazat celý účet.



## Použitá literatura

---

1. Node.js. *Wikipedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-01-14]. Dostupné z: <https://cs.wikipedia.org/wiki/Node.js>
2. React. *Wikipedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-01-14]. Dostupné z: [https://cs.wikipedia.org/wiki/React\\_\(webový\\_framework\)](https://cs.wikipedia.org/wiki/React_(webový_framework))
3. Express. *Wikipedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-01-14]. Dostupné z: [https://cs.wikipedia.org/wiki/React\\_\(webový\\_framework\)](https://cs.wikipedia.org/wiki/React_(webový_framework))
4. Mysql. *Wikipedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-01-14]. Dostupné z: <https://cs.wikipedia.org/wiki/MySQL>

## Seznam obrázků

---

Obrázek 1.....	8
Obrázek 2.....	9
Obrázek 3.....	9
Obrázek 4.....	9
Obrázek 5.....	10
Obrázek 6.....	10
Obrázek 7.....	10
Obrázek 8.....	11
Obrázek 9.....	11
Obrázek 10.....	11
Obrázek 11.....	11
Obrázek 12.....	12
Obrázek 13.....	12
Obrázek 14.....	13
Obrázek 15.....	13
Obrázek 16.....	13
Obrázek 17.....	14
Obrázek 18.....	14
Obrázek 19.....	14
Obrázek 20.....	15
Obrázek 21.....	15
Obrázek 22.....	15
Obrázek 23.....	16
Obrázek 24.....	16

## Obsah média

---

Zde přidejte stručně adresářovou strukturu (např jako víceúrovňový seznam) pro všechny důležité soubory. Je jasné, že pokud na médium (CD, DVD, Flashdisk) dáváte celý projekt s mnohými knihovnamí, nebudete zde vypisovat cesty ke všem souborům. Pouze navedete například kde se nachází projekt, kde se nachází build...

Médium by mělo být fyzicky označené **jménem, třídou, školním rokem!** Zároveň by médium mělo být v dokumentaci zajištěno tak, aby nevypadávalo, ale zároveň aby se dalo vyndat a použít.

Médium by mělo obsahovat následující:

- Projekt
- Případný export databáze
- Spustitelný build (nebo aspoň odkaz, kde se nachází spustitelná verze)
- Dokumentace v PDF + nějakém dalším editovatelném formátu (docx, odt...)
- Prezentace připravená k obhajobě

Backend

/script/items.js

/script/list.js

/script/login.js

/script/register.js

/script/sendMail.js //nefunkční a nepoužité kvůli googlu

/script/shop.js

/script/user.js

server.js //spustitelný soubor na zapnutí serveru

## FrontEnd

- /src/components/admin/AdminPanel.js
- /src/components/admin/EditLayout.js
- /src/components/admin/ShopLayout.js
- /src/components/list/AddItem.js
- /src/components/list/AddTypeToList.js
- /src/components/list/CreateList.js
- /src/components/list/EditItem.js
- /src/components/list/Item.js
- /src/components/list/List.js
- /src/components/list/SetPrice.js
- /src/components/mainpage/AddToFamily.js
- /src/components/mainpage/ForgotPassword.js //nefunkční a nepoužité kvůli googlu
- /src/components/mainpage/HeaderMainpage.js
- /src/components/mainpage/Login.js
- /src/components/mainpage/Register.js
- /src/components/profile/ShowProfile.js
- /src/components/profile/Statistika.js
- /src/components/Header.js
- /src/components/Mainpage.js
- /src/components/ShowList.js
- /src/App.js
- /src/index.css
- /src/index.js