

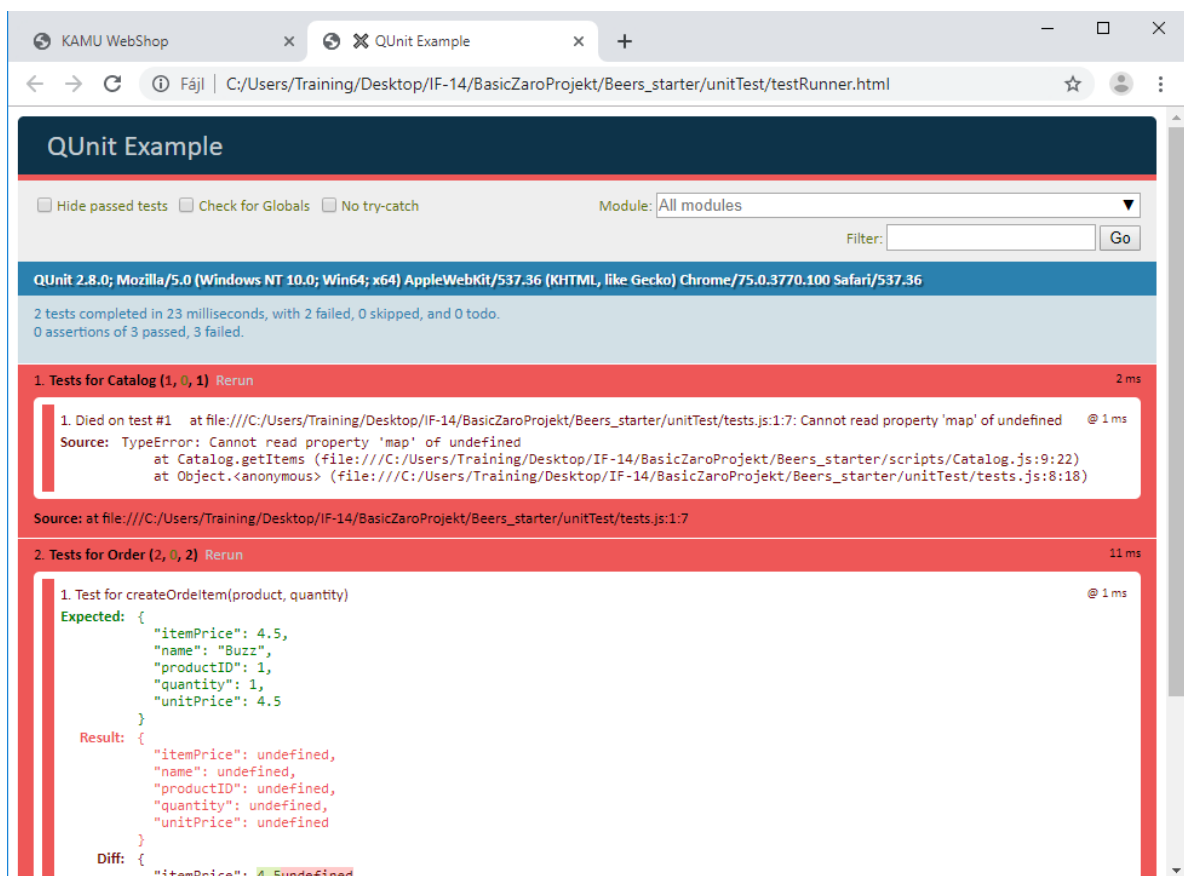
A záróvizsga kódíró feladataként egy részben már elkészült programot kell befejezni. A megoldás során törekedni kell a követelmények 100%-os teljesítésére. Egyes feladatokhoz részletes leírás áll rendelkezésre, míg más feladatokhoz csak rövid útmutató készült. Mindig el kell olvasni, meg kell érteni minden segítséget, útmutatást. A feladatok egymásra épülnek ezért azokat a megadott sorrendben kell megoldani, hogy szépen apránként elérjük az elvárt működést. Ha valami nagyon nem stimmel, akkor tessék használni a böngészőben rendelkezésre álló fejlesztői eszköztárt (console, debug, stb.).

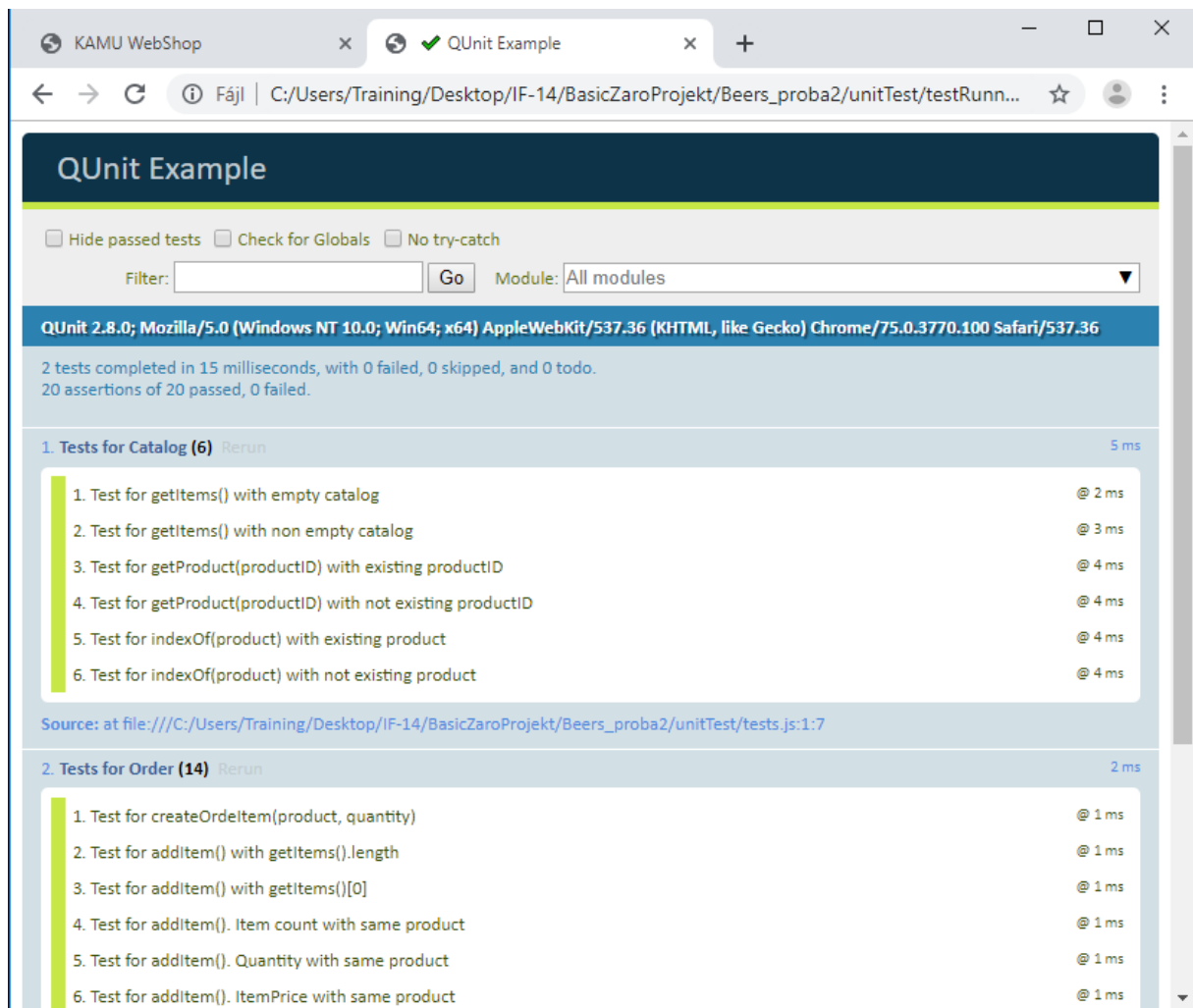
A befejezendő alkalmazás szimulál egy komoly háromrétegű alkalmazást. Megjelenik benne az adat~, az üzletlogikai~ és a megjelenítési réteg. Az alkalmazásban felhasznált adatok a *Beers/data/dataSource.js* állományban vannak. Érdemes belenézni, de módosítani nem kell. A feladatok megoldása során először az alkalmazásunk üzleti rétegét kell elkészítenünk. Ehhez nyújtanak támogatást az előre elkészített kódok és egység tesztek. Ez utóbbiakba bele vannak kódolva a követelmények. A következő nagyobb etap során az alkalmazásunk megjelenítési rétegében kell dolgoznunk.

Egységtesztekkel támogatott fejlesztés

A tesztek módosítása tilos (*Beers/unitTest/tests.js*).

Célunk: megfelelni az egységtesztekben rögzített elvárásoknak (unit test, QUnit környezetben). Nyissuk meg a Visual Studio Code fejlesztőeszközben a *Beers* foldert. A *Beers/unitTest/testRunner.html* állományt pedig Chrome böngészőben. Az alábbi ábrák közül az első azt szemlélteti, hogy elsőre mi tárul szemünk elé a böngészőben. A második ábra az elérendő célt mutatja. Azaz a megírt kódnak minden tesztesethez meg kell felelnie.





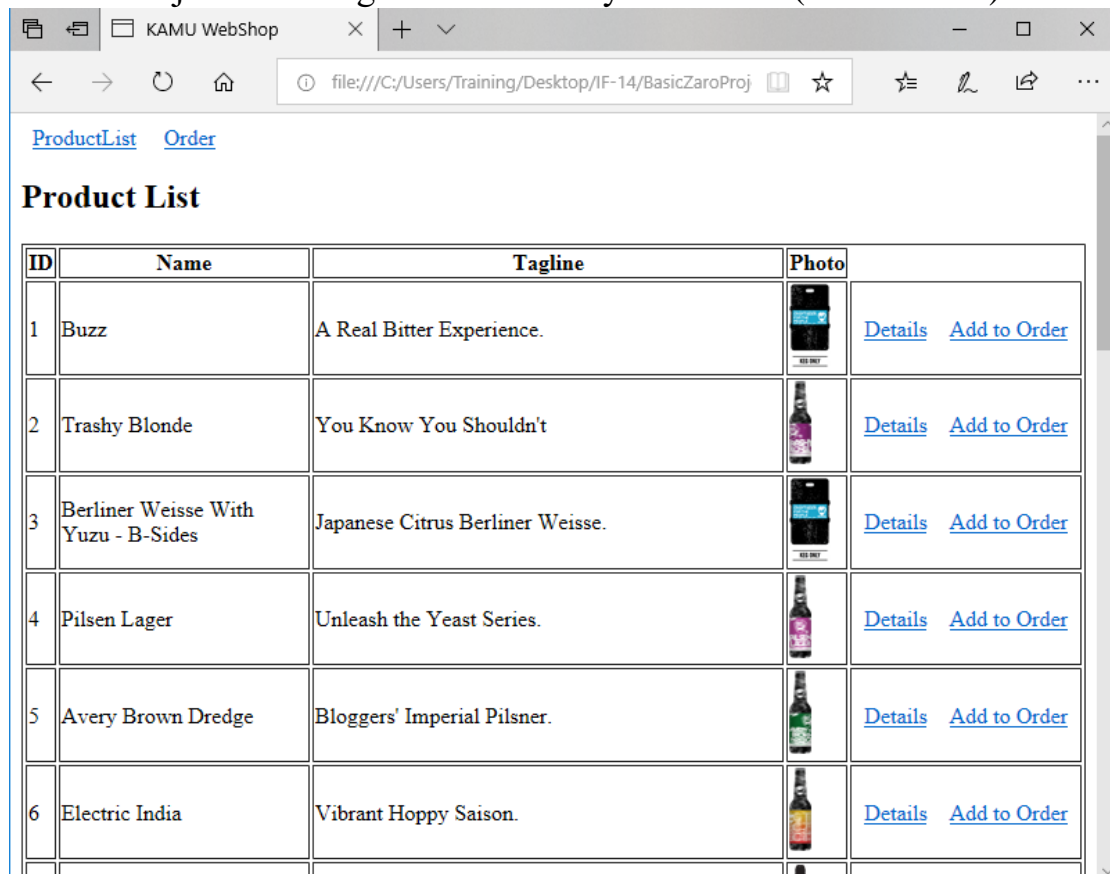
1. A rendelkezésre álló egységtesztek, illetve leírások alapján be kell fejezni a **Catalog** függvényt (*scripts/Catalog.js*).
 - a. az **items** változó értéke legyen a **products** argumentum értéke, vagy ha nincs argumentum, akkor üres tömb.
 - b. az **indexOf()** függvény vizsgálja az **items** változó tartalmát. Bemenete legyen egy product objektum. Kimenete pedig -1 ha nincs a katalógusban a keresett termék egyébként a tartalmazott termék indexe.
 - c. a **getProduct()** függvény vizsgálja az **items** változó tartalmát. Bemenetként várjon egy számot a **productId** argumentumból. Kimenete null vagy maga a tartalmazott product objektum.







2. A rendelkezésre álló egységtesztek, illetve leírások alapján be kell fejezni az `Order` függvényt (*scripts/Order.js*). Használd fel a `this.indexOf()` metódust, azért van.
- a. a `createOrderItem()` függvény létrehoz egy megrendelés tétel objektumot és visszaadja azt kimenetként. Bemenete egy termék objektum és a vásárolni kívánt mennyiség.
 - b. az `items` változó legyen egy üres tömb.
 - c. az `addItem()` hozzáadja az `items` nevű tömbhöz az `orderItem` paraméterként kapott megrendelés tételt. Ha a megrendelésben már van ilyen termék, akkor csak beállítja az adott tétel mennyiség és tételár értékét a paraméter alapján.
 - d. a `getItem()` függvény az `items` tömbből visszaadja a bemenetként megadott `productID`-jű terméket. Ha a megrendelésben nem található termék ilyen azonosítóval, akkor null értéket ad vissza.
 - e. a `getTotalPrice()` függvény kimenetként adja vissza a megrendelésben található tételek összértékét.
 - f. a `removeItem()` függvény segítségével lehet eltávolítani terméket a megrendelésből. Bemenete a termék azonosítója (`productID`).

Programozás szöveges követelmények alapján

Jelenítsük meg böngészőben az alkalmazást (*Beers/index.html*). Az alábbi leírásokat követve javítsuk, illetve készítsük el az alkalmazás megjelenítési rétegét. A beszúrt ábrák az elvárt megjelenést mutatják.

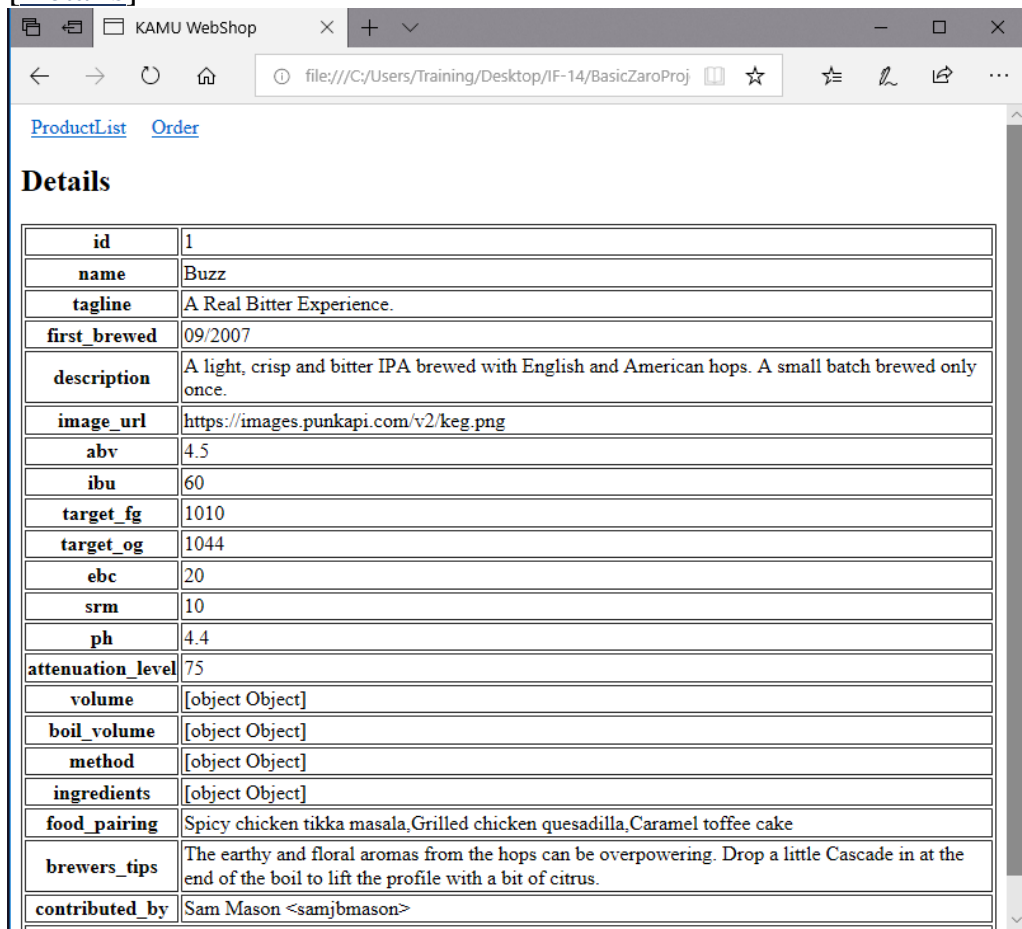
- Miért nem jelennek meg a termékek a nyitó oldalon (Product List)?



ID	Name	Tagline	Photo	
1	Buzz	A Real Bitter Experience.		Details Add to Order
2	Trashy Blonde	You Know You Shouldn't		Details Add to Order
3	Berliner Weisse With Yuzu - B-Sides	Japanese Citrus Berliner Weisse.		Details Add to Order
4	Pilsen Lager	Unleash the Yeast Series.		Details Add to Order
5	Avery Brown Dredge	Bloggers' Imperial Pilsner.		Details Add to Order
6	Electric India	Vibrant Hoppy Saison.		Details Add to Order

- A termékeket ábrázoló képek legyenek egységesen 60px magasak. A stílus beállítást a *design/main.css* állományban helyezzük el.
- Ismerkedjünk tovább az alkalmazás működésével.

- a. nézzük meg egy tetszőlegesen kiválasztott termék részletes leírását [Details]

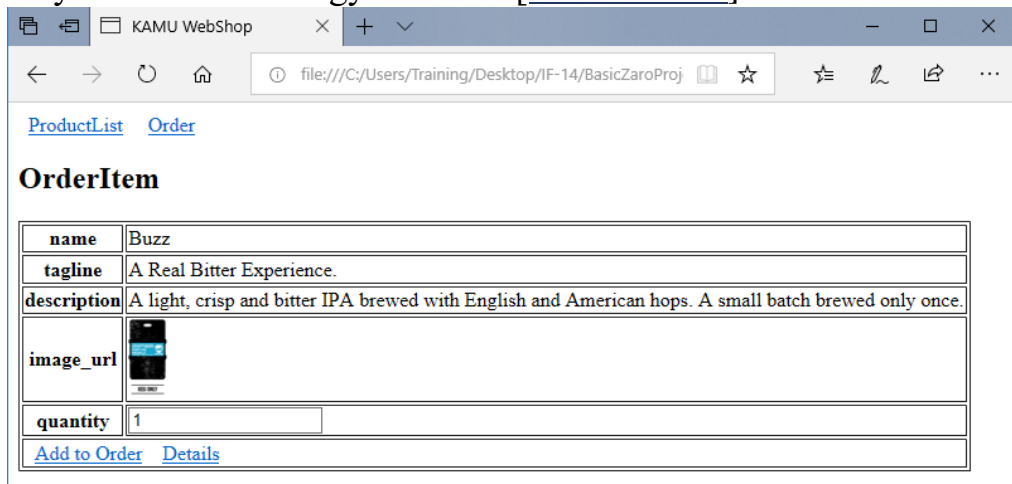


The screenshot shows a web browser window titled 'KAMU WebShop'. The address bar shows a file path. The page has two links: 'ProductList' and 'Order'. The main heading is 'Details'. Below it is a table with the following data:


id	1
name	Buzz
tagline	A Real Bitter Experience.
first_brewed	09/2007
description	A light, crisp and bitter IPA brewed with English and American hops. A small batch brewed only once.
image_url	https://images.punkapi.com/v2/keg.png
abv	4.5
ibu	60
target_fg	1010
target_og	1044
ebc	20
srm	10
ph	4.4
attenuation_level	75
volume	[object Object]
boil_volume	[object Object]
method	[object Object]
ingredients	[object Object]
food_pairing	Spicy chicken tikka masala, Grilled chicken quesadilla, Caramel toffee cake
brewers_tips	The earthy and floral aromas from the hops can be overpowering. Drop a little Cascade in at the end of the boil to lift the profile with a bit of citrus.
contributed_by	Sam Mason <samjbmason>

- b. térjünk vissza a termék katalógus oldalra [Product List]

- c. helyezzünk kosárba egy terméket [Add to Order]



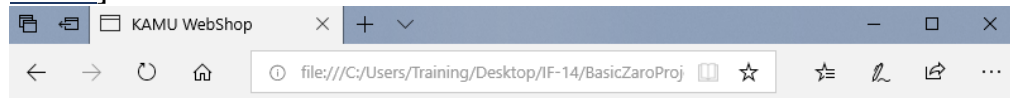
The screenshot shows the 'OrderItem' form in the KAMU WebShop. It contains the following fields:

name	Buzz
tagline	A Real Bitter Experience.
description	A light, crisp and bitter IPA brewed with English and American hops. A small batch brewed only once.
image_url	
quantity	<input type="text" value="1"/>

At the bottom, there are two links: 'Add to Order' and 'Details'.

- d. állítsuk be a venni kívánt mennyiséget

- e. adjuk a megrendeléshez a létrehozott megrendelés tételt [[Add to Order](#)]



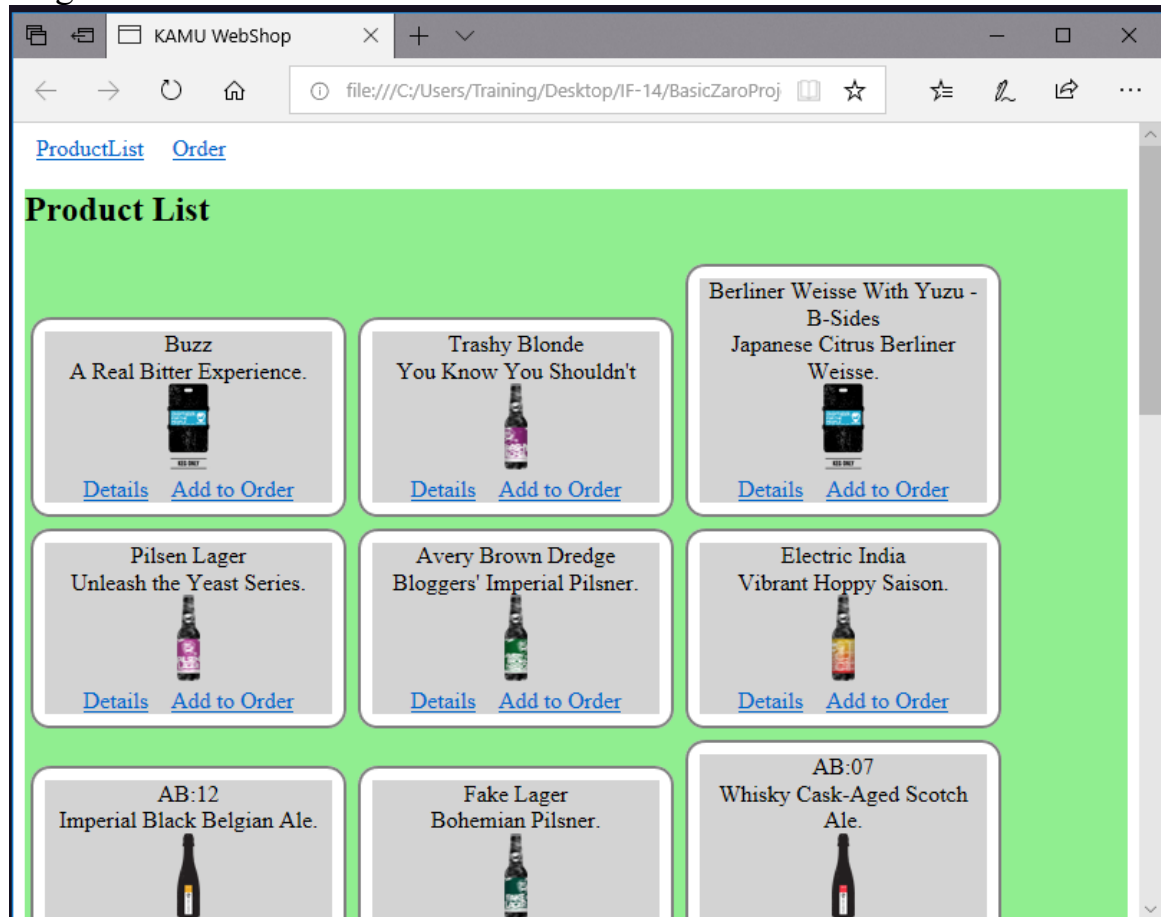
Order

ID	Product	UnitPrice	Quantity	ItemPrice	
1	Buzz	4.50	1.00	4.50	Edit Delete
2	Trashy Blonde	4.10	12.00	49.20	Edit Delete
Összesen				53.70	

- Miért nem jelennek meg a tételek a megrendelésben?
 - a. Az *scripts/dataVisualization/orderItemList.js* állományban a részletes útmutató szerint haladva gondoskodjunk a megrendelés tételek megjelenítéséről (vizualizáció). Üres sorok jelzik az utasítások helyét. Mindig a megjegyzést követő üres sorba kerüljön a megoldás. Itt minden változó, ami szükséges a sikerhez, előre deklarálva lett.

A következő feladatokat tetszőleges sorrendben lehet megoldani

- Alakítsuk át a termék listát úgy, hogy táblázat helyett kártyákon jelenjenek meg a termékek.



- CSS és grafikai elemek segítségével tegyük széppé, egységessé az oldalak kinézetét.
- Implementáljunk keresést a termékek nevére. A részleges egyezésre is működő keresés ne tegyen különbséget kis és nagybetűk között. A felsorolás után található képek segítenek a GUI elkészítésében. Plusz segítségként megnyitható a *Beers/unitTest/testRunner_2.html* állomány a böngészőben.
 - A [Catalog](#) függvénybe (*scripts/Catalog.js*) vegyünk fel egy kívülről nem elérhető [searchInfo](#) nevű változót alapértelmezetten null értékkel.
 - A [Catalog](#) függvénybe vegyünk fel egy kívülről elérhető [setSearch\(mitkeres\)](#) metódust. Ebben az új metódusban vizsgáljuk a [mitkeres](#) argumentum értékét, ha ez undefined vagy null vagy üres string, akkor állítsuk a [searchInfo](#) változó értékét null-ra. Egyéb esetekben a [searchInfo](#) változónak adjunk értékül egy anonymous objektumot `{ what: mitkeres }`.

- c. Bővítsük a `getItems()` metódust. Ha a `searchInfo` változó értéke nem null, akkor `result` legyen egyenlő a `result` megfelelően szűrt értékével. A keresésben alkalmazzuk a `searchInfo.what` tulajdonság értékét.
- d. Az `index.html` állományban írjuk meg az elvárt kinézethez szükséges HTML kódokat. Ne feledkezzünk meg a szövegdoboz és a gomb egyedi azonosításáról, hiszen később JavaScript-ből szeretnénk elérni őket.
- e. A `scripts/main.js` állományban a megfelelő helyen adjunk eseménykezelőt a [Search] gomb click eseményéhez. Az eseménykezelőben használjuk fel az alábbi kódrészletet, ahol a `searchWhat` változó értékét a szövegdobozból olvassuk be.

```
catalog.setSearch(searchWhat);  
Application.changeView('ProductList');
```

