

Áruház

Kurityák Dániel

Készítse el egy áruház webes bevásárlófelületét a leírásnak és a mintának megfelelően!

Előkészületek

1. Klónozza le a webprog alapot egy web, és a json-server alapot egy jsonserver mappába!
2. Másolja be a db.json fájlt a jsonserver/data/db.json helyére!
3. Másolja át az index.html fájlt a web mappába, ezzel kicserélve az alap csomagban található! Egészítse ki a fájlt a product-card.html és product-info.html fájlok tartalmával!
4. Másolja be az images mappát a src/assets mappába, valamint a favicon.ico fájlt cserélje a forrásban kapottra a public mappában!

Indítsa el a konténereket, majd vizsgálja meg az elérhető végpontokat!

Végpontok hívása

A REST API végpontok hívásáért külön fájlok lesznek a felelősek.

A végpontok hívásánál vegye figyelembe a következőket:

- Adja meg a kérések metódusát illetve a várt válasz típusát!
- Kezelje, hogy amennyiben a kérés sikertelen, úgy dobjon hibát! Amennyiben sikeres, úgy visszaadhatja a kérés törzsét!

products.js

5. Készítse el a products.js fájlt a src/js mappába!
6. Készítsen egy BASE_URL konstans változót, amiben legyen elérhető a jsonserver címe (pl. http://vm1.test:8888)
7. Készítsen és exportáljon egy getProducts() függvényt, amely a /products útvonalról lekéri az árucikkek adatait, majd azokat visszaadja egy promise-ban.
8. Készítsen és exportáljon egy getProduct(id) függvényt, amely a /products/{id} útvonalról lekéri a paraméterben kapott azonosítójú árucikk adatait, majd azt visszaadja egy promise-ban.

stores.js

9. Készítse el a stores.js fájlt a src/js mappába!
10. Készítsen egy BASE_URL konstans változót, amiben legyen elérhető a jsonserver címe (pl. http://vm1.test:8888)
11. Készítsen és exportáljon egy getStore(id) függvényt, amely a /stores/{id} útvonalról lekéri a paraméterben kapott azonosítójú áruház adatait, majd azt visszaadja egy promise-ban.

Kártyák generálása

12. Készítse el a `cards.js` fájlt a `src/js` mappába!
13. Készítsen és exportáljon egy `generateProductCard(product)` függvényt, amely paraméterül kapja egy árucikknek az adatait!

- a. Készítsen másolatot a `product-card` azonosítójú template-ből!
- b. Töltse ki a megfelelő elemeket az árucikk adataival!

| Elem osztálya | Kapcsolódó adat |
|--------------------------------|--|
| <code>.image</code> | <code>product.image</code> |
| <code>.name</code> | <code>product.name</code> |
| <code>.volume</code> | <code>product.volume</code> |
| <code>.price</code> | <code>product.price</code> |
| <code>.base-unit .unit</code> | <code>product.baseUnitPrice.unit</code> |
| <code>.base-unit .price</code> | <code>product.baseUnitPrice.price</code> |

- a. A kép alternatív szövege legyen az árucikk neve!
 - b. A két árat tartalmazó elemekben lévő adatok `toLocaleString()` segítségével legyenek a német területi beállításhoz megfelelően, euróban megjelenítve!


```
toLocaleString("de-DE", { style: "currency", currency: "EUR" })
```
 - c. Amennyiben a `product.baseUnitPrice.price` értéke null, úgy ahelyett ne jelenjen meg semmi!
 - d. A függvény adjon vissza a kitöltött kártyát!
14. A `main.js` fájlban készítsen egy `generateProductCards(products)` függvényt, amelynek paramétere az árucikkek tömbje. Menjen végig az árucikkeken és készítsen hozzájuk kártyákat, amiket adjon hozzá a `products-grid` azonosítójú elemhez!
 15. Szintén a `main.js` fájlban kérje le az összes árucikket, majd generálja le a promise által kapott adatokkal a kártyákat!

Árucikk felugró ablaka

16. Készítse el a `dialogs.js` fájlt a `src/js` mappába, amely a felugró ablakok megjelenítéséért felel.
17. Készítse el és exportálja az `openProductDialog(product)` függvényt!
 - a. Töltse ki a `#product-info` azonosítójú dialógus elemeit!

| Elem osztálya | Kapcsolódó adat |
|--------------------------------|--|
| <code>.image</code> | <code>product.image</code> |
| <code>.category</code> | <code>product.category</code> |
| <code>.brand</code> | <code>product.brand</code> |
| <code>.name</code> | <code>product.name</code> |
| <code>.volume</code> | <code>product.volume</code> |
| <code>.price</code> | <code>product.price</code> |
| <code>.base-unit .unit</code> | <code>product.baseUnitPrice.unit</code> |
| <code>.base-unit .price</code> | <code>product.baseUnitPrice.price</code> |

- a. A kép alternatív szövege legyen az árucikk neve!
- b. A két árat tartalmazó elemekben lévő adatok `toLocaleString()` segítségével legyenek a német területi beállításhoz megfelelően, euróban megjelenítve!


```
toLocaleString("de-DE", { style: "currency", currency: "EUR" })
```
- c. Jelenítse meg azokat a boltokat a nevét az `available-stores` azonosítójú listában, amelyekben az adott árucikk kapható (`product.availableStores`). A listaelemek kapján meg a `cursor-pointer` osztályt, valamint gondoskodjon arról, hogy a listaelemek új dialógus megnyitásakor törlődjenek!
- d. Nyissa meg a dialógust modal-ban!

18. Módosítsa a `cards.js` fájlban lévő `generateProduct()` függvényt úgy, hogy ha a felhasználó a kártyán belüli gombra nyom rá, úgy az adatok esetleges frissülése miatt kérje le az adott árucikket, majd hívja meg a friss adatokkal, amiket a `promise-on` keresztül kapott az `openProductDialog()`-ot!

Áruház felugró ablaka

19. Az előző dialógus mintájára készítsen egy új dialógust, melynek legyen `store-info` az azonosítója!
 - a. A dialógus az alábbi osztályokat kapja:

```
p-4 place-self-center border border-gray-300 w-[91.5%] max-w-150
```
 - b. Készítsen egy olyan űrlapot, amibe az elhelyezkedő gomb bezárja az űrlapot! A gomb kinézetéhez vegye alapul az előző dialógus hasonló gombját!
 - c. Jelenjen meg egy `h2` elem, ami az áruház típusát fogja megjeleníteni, a `type text-2xl font-bold` osztályokkal!
 - d. Továbbá legyen egy `p` elem, ami az áruház címének megjelenítésére szolgál, az `address text-xl` osztályokkal!
20. A `dialogs.js` fájlban készítsen egy `openStoreDialog(store)` függvényt, ami kitölti az előző feladat dialógusát és modal-ként megjeleníti azt!
21. Az `openProductDialog()` függvényt módosítsa úgy hogy amikor a felhasználó rákattint egy `li` elemre az elérhető áruházak közül, úgy az adatok esetleges frissülése miatt kérje le az adott áruházat, majd hívja meg a friss adatokkal, amiket a `promise-on` keresztül kapott az `openStoreDialog()`-ot!