JavaScript projektek

Az OOP nem elvárás, csak ahol ez külön ki van osztva szerepkörre!

1. Webáruház (7 ember)

Készítsetek egy webáruházat. Mivel a háttérben nem lesz adatbázis, sem szerveroldali nyelv, ezért ez csak egyfajta szimuláció lesz. Nincs regisztráció és felhasználókezelés! A modulok:

- **Design és struktúra (2 ember):** a HTML és CSS elkészítése, a JS kódok összefésülése, az oldalak között a kapcsolat kialakítása. Pl.: süti (cookie) vagy webstorage segítségével.
- **Termékek (1 ember):** lehessen termékeket feltölteni egy űrlapon keresztül, amiket egy tömbben tárolunk el. A tömböt érdemes sütibe vagy webstorage-ba menteni, hogy később listázható legyen más oldalakon. Adatok a termékeknél:
    - terméknév
    - darabszám
    - cikkszám
    - ár
    - termékleírás
- **Terméklista (1 ember):** a termékek oldalon kilistázni a felvitt termékeket egy tömbből, melyek a designba illeszkedő módon jelennek meg az oldalon. Nem kell készíteni termékrészlet oldalt, csak egyetlen oldal kell, ahol listázásra kerülnek maguk a termékek. Itt a termékeknél jelenjen meg egy mintakép (pl.: placeholder), de akár generálhatunk is meglévő képek közül véletlenszerűen egy IMG HTML elem src attribútumába egy mappából. Legyen minden terméknél egy INPUT, ahol egy NUMBER típusú léptetővel lehessen kiválasztani a mennyiséget. Csak annyit lehessen kiválasztani, amennyi van raktáron. Legyen egy KOSÁRBA gomb, amivel el lehet menteni egy tömbbe a kosárba tett termékeket. Ha nincs készleten a termék, vagy beleraktuk az összeset, akkor legyen inaktív a KOSÁRBA gomb.
- **Kosár (2 ember):** a kosárnak a kilistázása a tömbből, majd a designnak megfelelően a DOM-ba illesztése. Itt szintén lehessen még módosítani a kosárban lévő termékek számát (növelni ne lehessen, csak csökkenteni). Ehhez legyen egy INPUT szövegmező, amibe lehessen beírni számot, de csak kevesebbet, mint ami már benne van. Legyen egy frissít gomb, amit ha megnyomnak, írja át a kosárban lévő termékek számát. Ha az INPUT-ba 0-át írunk és úgy frissítjük, töröljük ki automatikusan a terméket, illetve legyen egy törlés gomb, amivel szintén töröljük a DOM-ból a termék sorát. Minden terméknél legyen egy egységár is, amit a darabszámmal megszorozva jelenítsünk meg tételenként. Végül a termékek alatt legyen egy végösszeg is, ahol szintén kiszámoljuk az összértéket. Legyen egy megrendelés gomb, amire rákattintva törlődik a kosár tartalma.
- **Kereső (1 ember):** legyen egy kereső, ami a termékek tömbjében rákeres a mezőbe beírt szövegre vagy szövegrészletre. A találati lista egy új oldalon jelenjen meg formázva.

2. Online kölcsönző (8 ember)

Készítsetek egy online könyvtár programot, ami ahhoz hasonló, amit az órán csináltunk, csak ez grafikus felülettel rendelkezik. A modulok:

- **Design és struktúra (2 ember):** a HTML és CSS elkészítése, a JS kódok összefésülése, az oldalak között a kapcsolat kialakítása. Pl.: süti (cookie) vagy webstorage segítségével.
- **Belépés oldal (1 ember):** a kezdőlapon kérjen be az oldal egy e-mail címet és egy jelszót. Regisztráció NINCS! Előre legyen létrehozva 5 belépési adat. A belépési adatokat egy sütiből / web storage-ból olvassa ki a program. Ha szerepel ilyen a tömbbe, akkor beengedjük, ha nem, akkor hibaüzenetet írunk ki neki. A program 5 próbálkozást engedélyezzen! Ehhez vegyünk fel egy új süti adatot, ami 5 percig blokkolja a user, azaz a süti 5 percig ne törlődjön ki. A süti a gépen tárolódik, így csak akkor tudja a tiltást feloldani, ha kitörli a böngészőből. Sikeres belépéskor irányítson át egy új oldalra, ahol lehessen hozzáférni a user sütiben tárolt adataihoz.
- **Új könyv feltöltése (1 ember):** egy űrlap segítségével lehessen új könyveket feltölteni, melyeket egy tömbbe mentünk, ami átkerül majd egy sütibe / web storage-ba. A könyvhöz képet úgy lehessen megadni, hogy a fájlnevet kell majd beírni egy INPUT mezőbe, ezt fogja kilistázni egy IMG elembe a kezdőlapon az adott könyvhöz egy mappából. A következő adatokat legyen kötelező megadni:
    - könyv címe
    - ISBN száma
    - író(k)
    - könyv borítójának fájlneve
    - 1-2 mondatos leírás
- **Könyvek oldal (2 ember):** itt legyenek kilistázva a kölcsönözhető könyvek. Nem kell a könyveknek részlet aloldalt csinálni, elegendő egy oldalon kilistázni mindet. Minden könyvből csak egy példány legyen, hisz egynél többet úgyse kölcsönzünk belőle. A készletkezelést nem kell emiatt megoldani. Akkor tudjon valaki könyvet kölcsönözni, ha rákattint a „kölcsönzés” gombra. Egy user maximum 3 könyvet kölcsönözhessen. Ha ezt meghaladja, akkor az összes gomb inaktív legyen. A kikölcsönzött könyveknél a gomb felirata legyen „visszavétel”, amit ha megnyomunk, automatikusan váltson vissza „kölcsönzés” feliratra és így a funkciója is változzon vissza. Az oldal tetején valahogy jelöljük, hogy jelenleg hány db. könyv van nálunk pl.: egy számmal, ami dinamikusan változzon, ha visszahozunk / kölcsönzünk újabb könyvet. Rögzítsük a kölcsönzés pontos időpontját dátum és időpont alapján! Ne kölcsönözhessen egyetlen könyvet se az, akinek van lejárt kölcsönzése!
- **Admin oldal (2 ember):** ezt az oldalt a könyvtáros tudja majd kezelni. Itt legyenek kilistázva a felhasználók (5 user lesz). Ezeket sütiből / web storage-ból kell majd kiolvasni. A usereknek a neve lehessen lenyitható, lenyitáskor pedig jelenjen meg egy lista, ami a részleteket jeleníti meg. Ha a kölcsönzés időpontja túllépi a 2 órát (a szimuláció miatt), akkor pirossal jelenjen meg. Az adatoknál ezek szerepeljenek:
    - e-mail cím
    - név
    - kölcsönzött könyvek címe, ISBN száma, kis borítóképe
    - kölcsönzés időpontja
    - büntetés díja (100 Ft / nap)

3. Partiszervező (6 ember)

Szintén az órai feladatra épít, ám a funkciók jobban kibővülnek. A modulok:

- **Design és struktúra (2 ember):** a HTML és CSS elkészítése, a JS kódok összefésülése, az oldalak között a kapcsolat kialakítása. Pl.: süti (cookie) vagy webstorage segítségével.
- **Ember hozzáadása (1 ember):** a meghívottnak a nevét kelljen megadni egy INPUT-ban, amit ne lehessen üresen beküldeni a DOM-ba. Minden meghívott kapjon egy egyedi azonosítót, ami jelenjen meg a neve alatt pl.: mint Discordnál (Kornél#4982). Minden ember kapjon véletlenszerűen egy profilképet (pl.: egy mappában lévő képek közül véletlenszerűen egy fájlnevet választunk ki és azt illesztjük be az IMG src attribútumába), tehát képet nem kell feltölteni. Jelenjen meg egy formázott dátum is, ami piciben mutatja a meghívás időpontját magyar formátumban: pl.: 2020. 04. 01. 16:33:21. Létező azonosítóval és névvel ne lehessen még egy embert felvinni.
- **Ember szerkesztése (1 ember):** lehessen az adott meghívott nevét szerkeszteni. Lehessen hozzáadni maximum 3 db. új megjegyzést. Egy megjegyzés egy INPUT mezőt jelent. A meglévő megjegyzéseket lehessen módosítani és törölni is. Pl.: telefonszámot, e-mail címet, honnan ismerjük stb. Ha kimerítettük a 3 megjegyzést, egyszerűen ne lehessen újat hozzáadni, csak ha törlünk belőle. A meghívott kártyáját lehessen törölni. Továbbá lehessen jelölni, hogy az illető megerősítette-e a szándékát.
- **Kereső (1 ember):** legyen az oldalon egy kereső, ami tudjon szűrni névre, e-mail címre, illetve megerősített szándékra. A kártyákat kellene itt megjeleníteni ill. elrejteni.
- **Fejlesztő (1 ember):** gondoskodik a kód OOP-ben történő megvalósításról, az osztályok létrehozásáról és a csoport többi tagja által gyártott kódsor beépítéséről.

4. Tanulmányi rendszer (kis Neptun) (7 ember)

Készítsünk egy egyszerűsített Neptun rendszert. A modulok:

- **Design és struktúra (2 ember):** a HTML és CSS elkészítése, a JS kódok összefésülése, az oldalak között a kapcsolat kialakítása. Pl.: süti (cookie) vagy webstorage segítségével.
- **Bejelentkezés (1 ember):** legyen egy login oldal, ahol sütiből / web storage-ból kiolvasott adatok alapján lehessen belépni. Tároljuk le 3 különböző hallgató adatait. Az adatokat egy TXT fájlból olvassuk be, mielőtt azt elmentenénk sütibe vagy webstorage-ba. Ha szerepel ilyen a tömbbe, akkor beengedjük, ha nem, akkor hibaüzenetet írunk ki neki. A program 5 próbálkozást engedélyezzen! Ehhez vegyünk fel egy új süti adatot, ami 5 percig blokkolja a user, azaz a süti 5 percig ne törlődjön ki. A süti a gépen tárolódik, így csak akkor tudja a tiltást feloldani, ha kitörli a böngészőből. Sikeres belépéskor irányítson át egy új oldalra, ahol lehessen hozzáférni a user sütiben tárolt adataihoz. Az adatok:
    - NEPTUN kód
    - jelszó
    - szak
- **Neptun admin oldal (1 ember):** lehessen kurzusokat létrehozni az alábbi adatokkal, illetve módosítani és kitörölni őket. A kurzusokat szintén tároljuk el tömbként, ami egy sütiből / web storage-ból kerül kiolvasásra betöltéskor, majd elmentésre mentéskor. Egy kurzust csak akkor lehessen törölni, ha azon nincs hallgató! Ehhez az oldalhoz nem kell belépési adat meg user kezelés sem! Az adatok:
    - kurzus neve
    - kurzuskód
    - kreditszám
    - max. létszám (minimálisat adjunk meg a teszteléshez pl.: 2-3 fő)
    - oktató neve
    - vizsgás / gyakorlati jegyes
- **Hallgatói oldal (2 ember):** a bejelentkezett hallgató lássa az admin által létrehozott kurzusokat (ha hozott már létre). Egy hallgató max. 30 kreditnyi kurzust vehessen fel. Csak arra lehessen jelentkezni, amin van hely. A hallgató tudja megnézni a kurzus adatait, jelentkezéskor pedig jelenjen meg az ő neve is. Tudja leadni is a kurzust! A felvett kurzusokat valahogy jelöljük a hallgatónak, illetve jelenítsük meg az oldalon az összkreditszámot is! Az oldal betöltésekor ellenőrizzük, hogy kivel léptünk be a süti / web storage alapján és csak annak az adatai módosulhassanak.
- **Fejlesztő (1 ember):** gondoskodik a kód OOP-ben történő megvalósításról, az osztályok létrehozásáról és a csoport többi tagja által gyártott kódsor beépítéséről.

5. Kő-papír-olló játék (5 ember)

Készítsünk egy grafikus felületen játszható kő-papír-olló játékot, melyben a gép ellen lehet játszani.

- **Design és struktúra (1 ember):** a HTML és CSS elkészítése, a JS kódok összefésülése, az oldalak között a kapcsolat kialakítása. Pl.: süti (cookie) vagy webstorage segítségével.
- **Játékmenet (2 ember):** a játékosnak meg kelljen adnia a nevét (a nevéhez generáljunk automatikusan egy 4 jegyű számot, mint discord-on pl.: Kornél#8473), vagy kérhet névgenerálást is. Ha nevet generál, maximum ezt 3x tehesse meg, a 3. lehetőséget muszáj elfogadnia. Két egyforma név#szám user nem lehet! A játék során a gép is gondol egy elemre (kő, papír vagy olló), majd a játékosnak is választania kell egyet (pl.: listából / kép alapján rákattint stb.). Ezután egy gombot megnyomva jelenjen meg a gép és a játékos tippje, majd írjuk ki hogy ki nyert. A játékos és a gép pontszámait gyűjtsük folyamatosan. A játék addig tart, amíg ezt a játékos nem jelzi egy vége gombbal. Minden névmegadáskor és játszáskor új süti / web storage elem jöjjön létre, tehát a meglévő adatokat csak statisztikához lehessen felhasználni, játszani nem.
- **Statisztikák (1 pont):** minden játék végén sütibe / web storage-ba tároljuk el a játékos és a számítógép által elért pontokat ill. a játszott körök számát. Ez jelenjen meg grafikusan is egy listában, ahol legyenek a helyezettek sorba rendezve pontszám alapján. A számítógép egy generált nevet kapjon pl.: Computer#3942. A statisztika egy új oldalon legyen, ahová a játékmenet oldaláról lehet eljutni.
- **Fejlesztő (1 ember):** gondoskodik a kód OOP-ben történő megvalósításról, az osztályok létrehozásáról és a csoport többi tagja által gyártott kódsor beépítéséről.