Autó Konfigurátor Webes Alkalmazás Dokumentáció

Az Autó Konfigurátor Webes Alkalmazás egy olyan online eszköz, amely lehetővé teszi a felhasználók számára, hogy testre szabják a gépkocsijukat az adott gyártó által kínált különféle opciók és konfigurációk szerint. Az alkalmazás Vue.js és Node.js technológiákkal készült, amelyek lehetővé teszik a felhasználói felület gyors és interaktív betöltését és a visszajelzések késleltetés nélküli megjelenítését.

Az alkalmazás alapvetően három fő részre osztható: a frontend (felhasználói felület), a backend (szerveroldali logika) és az adatbázis.

Frontend

Az alkalmazás frontend része a Vue.js keretrendszerrel készült, amely lehetővé teszi a komponens alapú architektúra használatát. A frontend szerepe az, hogy megjelenítse a felhasználók számára a gépkocsik és a hozzájuk tartozó konfigurációs lehetőségek listáját, valamint lehetővé tegye a felhasználók számára, hogy válasszanak ezek közül. Az alkalmazás használja a Vue.js állapotkezelő funkcióit, amelyek lehetővé teszik az alkalmazás állapotának megőrzését a felhasználói munkamenet során.

Backend

Az alkalmazás backend része a Node.js keretrendszerrel készült, amely lehetővé teszi az API-k fejlesztését. Az API-k használatával a frontend kommunikál a szerverrel, és elkéri az adatokat az adatbázisból. A backend feladata az adatok feldolgozása, az üzleti logika implementálása, a hibakezelés és az adatbázis-kezelés. Az alkalmazás a Node.js Express keretrendszerét használja a HTTP kérések kezelésére és az API-k fejlesztésére.

Adatbázis

Az alkalmazás adatbázisa MySQL-rel van megvalósítva. Az adatbázisban tároljuk az összes elérhető autót és azok konfigurációs lehetőségeit. Az adatbázis-kezelést az alkalmazás backend része végzi.

Az alkalmazás működése A felhasználó az alkalmazás főoldalán találja magát, ahol a választható autók listája található. A felhasználó kiválaszthatja az általa kívánt autót, majd a konfigurációk oldalra irányítjuk át, a konfigurációk oldalon a felhasználó testre szabhatja az autót, kiválaszthatja a motor típusát, a karosszéria színét, az ülések anyagát, az infotainment rendszert stb. Az alkalmazás a felhasználó választásait folyamatosan frissíti és megjeleníti az árakat, így a felhasználó mindig tisztában van a választásainak költségeivel.

Miután a felhasználó befejezte a konfigurációt, az alkalmazás lehetővé teszi az autó megrendelését, amelyet a rendszer továbbít az ügyfélszolgálatnak. Az alkalmazás az autó kiválasztásának és a konfigurációjának az előzményeit is eltárolja, hogy a felhasználó bármikor visszatérhessen a korábbi konfigurációkhoz, vagy folytathassa az előzőleg félbehagyott konfigurációkat.

Az alkalmazás az ügyfelek számára is lehetővé teszi az autók raktárkészletének ellenőrzését, valamint az adott modellre vonatkozó aktuális akciók, kedvezmények és kiegészítő ajánlatok megtekintését.

Az alkalmazás felülete intuitív és könnyen használható, a rendszer pedig gyors és megbízható működést biztosít. Az alkalmazás egyszerűen testreszabható az adott gyártó termékeinek megfelelően, és lehetővé teszi a felhasználók számára, hogy az interneten keresztül kényelmesen és könnyen válasszák ki az álmaik autóját.

Összefoglalva, az Autó Konfigurátor Webes Alkalmazás egy modern és hasznos eszköz az autóvásárlók számára, amely a Vue.js és Node.js technológiákkal, valamint a MySQL adatbázissal kombinálva lehetővé teszi a felhasználók számára, hogy könnyen testre szabhassák az autójukat az adott gyártó által kínált opciók szerint.

Igen, természetesen. Az Autó Konfigurátor Webes Alkalmazásban a Vue.js alkalmazás a felhasználói felületet kezeli, míg a Node.js szerver oldali alkalmazás felelős a logika és az adatok kezeléséért. A két alkalmazást a dataservice.js fájl kapcsolja össze, amely az adatokat és funkciókat hozzáférhetővé teszi a Vue.js alkalmazás számára.

A dataservice.js fájlban definiáljuk az API kérések funkcióit a Node.js szerverhez, például az autókonfiguráció mentését, az autó kiválasztását, az árak számítását és az aktuális raktárkészlet lekérdezését. Ezeket a funkciókat a Vue.js alkalmazásban a dataservice objektumon keresztül érjük el.

Ezenkívül a dataservice.js fájl definiálja a felhasználói bejelentkezést és regisztrációt kezelő funkciókat is. Ezeket a funkciókat szintén a dataservice objektumon keresztül érjük el a Vue.js alkalmazásban.

A dataservice.js fájlban továbbá definiáljuk a Node.js szerverhez történő kapcsolódás paramétereit is, például az API végpont URL-jét és a hitelesítési adatokat.

A Vue.js alkalmazásban a dataservice.js fájlt a komponensekben importáljuk, és a dataservice objektumon keresztül hívjuk meg a szükséges funkciókat. Az adatok frissítéséhez a Vue.js komponensek a dataservice objektum segítségével történő aszinkron API hívásokat használnak, és a visszatérő adatokat az alkalmazás állapotában tárolják.

Az Axios-t használva a dataservice.js fájlban definiáltunk különböző funkciókat az adatok kezelésére és a Node.js szerver API-jával történő kommunikációra. Az Axios kérés küldése és a válasz fogadása a funkciók egyetlen sorában történik, ami nagyon hatékony és olvasható kódot eredményez.

Összefoglalva, a dataservice.js fájl a backend és frontend összekapcsolását valósítja meg az Autó Konfigurátor Webes Alkalmazásban. A Node.js szerver API-ját használva teszi elérhetővé az adatokat és funkciókat a Vue.js alkalmazás számára.

Természetesen. Az Autó Konfigurátor Webes Alkalmazásunkhoz MySQL adatbázist használtunk, amelyben számos tábla található. Az alábbiakban felsoroljuk az adatbázisban szereplő fontosabb táblákat és kapcsolótáblákat:

cars: Az autók adatait tartalmazó tábla. Minden sor egy autó adatait tartalmazza, például az autó típusát, modelljét, évjáratát, motorját és árát.

extras: Az extrák adatait tartalmazó tábla. Minden sor egy extrát tartalmaz, például navigációs rendszert, bőrüléseket, vagy ülésfűtést.

payment: A fizetési módokat tartalmazó tábla. Minden sor egy fizetési módot tartalmaz, például készpénzt, bankkártyát, vagy részletfizetést.

salons: Az autószalonok adatait tartalmazó tábla. Minden sor egy autószalon adatait tartalmazza, például a nevét, címét, és elérhetőségeit.

users: A felhasználók adatait tartalmazó tábla. Minden sor egy felhasználó adatait tartalmazza, például a nevét, email címét, jelszavát.

A táblák közötti kapcsolatokat több kapcsolótábla segítségével valósítottuk meg. Az alábbiakban felsoroljuk ezeket:

users\_payment: A felhasználók és fizetési módok kapcsolatát tartalmazó tábla. Minden sor egy felhasználó és egy fizetési mód kapcsolatát tartalmazza.

cars\_extras: Az autók és extrák kapcsolatát tartalmazó tábla. Minden sor egy autó és egy extra kapcsolatát tartalmazza.

cars\_salons: Az autók és autószalonok kapcsolatát tartalmazó tábla. Minden sor egy autó és egy autószalon kapcsolatát tartalmazza.

Ezek a kapcsolótáblák lehetővé teszik, hogy az alkalmazás az adatbázisban tárolt információkat összekapcsolja, és lehetővé tegye például, hogy egy felhasználó válasszon egy autót az autók közül, majd kiválassza a fizetési módot, majd a választott autót egy adott autószalonban vehesse át.

A cars tábla a következő oszlopokat tartalmazza:

id: Az autó egyedi azonosítója, amely egy egész szám típusú érték.

userid: Az autó tulajdonosának azonosítója, amely egy egész szám típusú érték. Ez egy külső kulcs, amely hivatkozik a users tábla id oszlopára.

model: Az autó modellje, amely egy szöveges típusú érték.

motor: Az autó motorja, amely egy szöveges típusú érték.

color: Az autó karosszériájának színe, amely egy szöveges típusú érték.

interiorcolor: Az autó belső tér színének, amely egy szöveges típusú érték.

Ezenkívül a kapcsolótáblákban szereplő cars\_extras és cars\_salons táblák is hivatkoznak az id oszlopra, amely az autók egyedi azonosítója. Ezek a kapcsolótáblák teszik lehetővé, hogy az alkalmazás az adatbázisban tárolt információkat összekapcsolja, és lehetővé tegye például, hogy egy felhasználó válasszon egy autót az autók közül, majd kiválassza a kívánt extra opciókat, vagy hogy egy adott autószalonban mely autók érhetőek el.

A users tábla az alkalmazás felhasználóinak adatait tárolja. Az oszlopok a következők:

id: Az egyedi azonosítója a felhasználónak, amely egy egész szám típusú érték.

name: A felhasználó teljes neve, amely egy szöveges típusú érték.

email: A felhasználó e-mail címe, amely egy szöveges típusú érték.

password: A felhasználó jelszava, amely egy szöveges típusú érték. Az adatbázisban tárolt jelszavakat általában hash-elve tárolják, hogy biztonságosabbak legyenek.

birthdate: A felhasználó születési dátuma, amely egy dátum típusú érték.

address: A felhasználó címe, amely egy szöveges típusú érték.

Az id oszlop az egyedi azonosító, amelyet a kapcsolatok létrehozásához használunk más táblákkal, például a cars vagy a users\_payment táblákkal. Az email és a password oszlopokat azonosításra és hitelesítésre használjuk a felhasználó bejelentkezésekor. Az address oszlop a felhasználó címét tartalmazza, amely például szállítási információk megadására hasznos. Az birthdate oszlop pedig a felhasználó születési dátumát tartalmazza, amely egyéb funkciókban, például az életkor kiszámításában is felhasználható.