STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor č. 18: Informatika

Digitální garáž

Pavel Pel

Pardubický kraj Pardubice
2021

STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor č. 18: Informatika

Digitální garáž

Autor: Pavel Pel
Škola: DELTA – Střední škola informatiky a ekonomie, s.r.o.
Ke Kamenci 151, 530 03 Pardubice
Kraj: Pardubický kraj
Konzultant: Bc. Vlaďka Janů

Pardubice 2020

Prohlášení

Prohlašuji, že jsem svou práci SOČ vypracoval/a samostatně a použil/a jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů.

Prohlašuji, že tištěná verze a elektronická verze soutěžní práce SOČ jsou shodné.

Nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších předpisů.

V Pardubicích dne 10. ledna 2021 ……

Pavel Pel

Poděkování

Děkuji Bc. Vlaďce Janů za dohled, vedení práce a za veškeré rady, které mi pomohly vypracovat tento projekt. Dále chci poděkovat svým kamarádům za podporu a testování finální verze projektu.

Anotace

Cílem tohoto projektu je vytvořit webovou aplikaci („Digitální garáž“), do které si můžete nahrát své auto a nechat ostatní inspirovat se a kochat. Hlavní částí je webová aplikace pro uživatele, která je plně responzivní na mobilních zařízeních. Projekt cílí na fanoušky automobilů a jejich úprav.

Klíčová slova

webová aplikace; Vue.js; Nuxt.js; Firebase;

Obsah

1 Úvod

Projekt Digitální garáž je nová platforma na trhu, je to v podstatě tematická sociální síť pro nadšence automobilů a jejich úprav, ať už vzhledových (interiér/exteriér), tak i výkonnostních (motorové/podvozkové úpravy). Aplikace tedy umožní každému uživateli přidat si své auto do své digitální garáže nebo se nechat inspirovat mezi automobily ostatních.

2 Inspirace projektu

Podnět pro vytvoření aplikace CarRate je inspirován projekty:

- Celosvětově známá sociální síť **Facebook**
- Inzertní server **Bazoš** a jeho plně funkční bez přihlašovací koncept

3 Technologie

3.1 NuxtJS

NuxtJS je framework pro Vue, který podporuje server-side rendering. Stránka se nejprve vykreslí na serveru a poté se teprve posílá klientovi na zařízení. Aplikace se díky tomu načítá podstatně rychleji, protože se nemusí nejprve stáhnout a načíst javascript, a až poté obsah stránky.

Zároveň jsou weby se server-side renderingem přívětivější pro vyhledávače v rámci SEO (optimalizaci pro vyhledávače), jelikož se jim posílá již vykreslený obsah.

Další vlastností NuxtJS je automatický routing. Všechny soubory, které jsou ve složce „/pages“ se automaticky označují jako separátní stránky, díky čemuž není potřeba používat externí router.

NuxtJS také automaticky optimalizuje build. Stránky se vykreslují vždy jen s knihovnami a JavaScriptem, který skutečně potřebují. NuxtJS tedy celý kód rozbíjí na menší celky, které poté importuje.

Mimo jiné NuxtJS také podporuje Hot Code Reloading – při vývoji se stránka přednačte automaticky pokaždé, když je v kódu zjištěna jakákoliv změna. (1)

3.2 Firebase

Firebase je platforma vyvinutá společností Google pro vývoj mobilních a webových aplikací. Obsahuje řadu klientských knihoven pro mobilní platformy (iOS, Android) i pro web (JavaScript). Kromě real-time databáze nabízí i možnost zasílání push notifikací, sledování statistik používání a mnoho dalšího. (2)

3.3 Vercel

Vercel je cloudová platforma pro nasazení projektu bez nutnosti vlastního serveru. Umožňuje vývojářům hostovat webové stránky a webové služby, které se nasazují

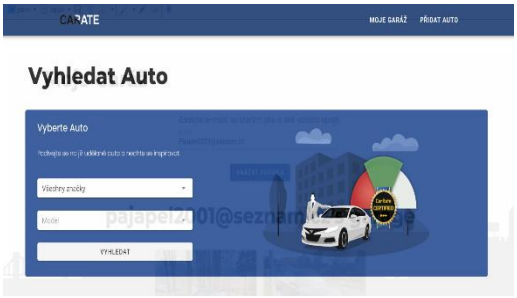
okamžitě, automaticky se mění a nevyžadují žádný dohled, to vše s minimální konfigurací. (3)

4 Funkce aplikace

4.1 Vyhledávání příspěvků

Vyhledání příspěvku jde docílit dvěma způsoby.

První možnost je vyhledávání na hlavní stránce, které má 3 druhy vyhledávání:



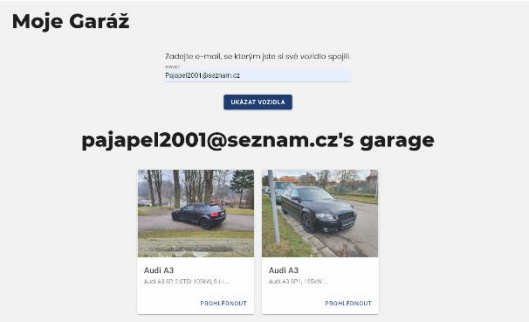
Obrázek 1 - Vyhledat Auto

- **Všechna vozidla** – zobrazí se všechna vozidla seřazená chronologicky
- **Vyhledání podle značky** – zobrazí se vozidla od požadované značky
- **Zadání přesného modelu** – zobrazí jen konkrétní vozidla s modelovým označením

Stránka s výsledky je přehledně uspořádaná pravidelná síť náhledových karet, které jsou rozřazeny po malém množství do jednotlivých stránek, mezi kterými se dá jednoduše přepínat ve spodní části sekce.

Vyhledávání je technologicky navrženo tak, že značka i model se vloží do URL adresy za lomítko, odkud si poté stránka s výsledky bere data a dále je zpracovává. Díky Nuxt.js se dá lehce docílit získání parametrů z adresy pomocí funkce „this.$route.params“.

Druhá možnost je vyhledávání na stránce „Moje garáž“, která dokáže vyhledat všechny příspěvky, které jsou spojeny s e-mailovou adresou, kterou zadáte do pole vyhledat a následně se vám ukáže vaše digitální garáž.



Obrázek 2 - Moje garáž

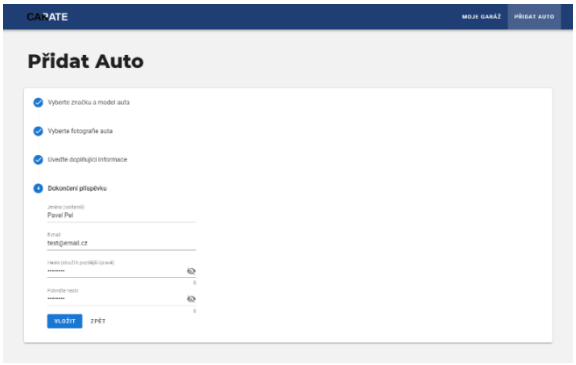
4.2 Přidání příspěvku

Tato stránka slouží k přidávání příspěvků (automobilů) do své digitální garáže.

Celým procesem vás provede jednoduchý krokový formulář, který se skládá z těchto částí:

1) Výběr konkrétní značky a modelu
2) Nahrání fotografií auta
3) Vypsání veškerých informací o automobilu
4) Uvedení potřebných údajů
   a) Jméno majitele
   b) E-mail
   c) Heslo

Po vyplnění formuláře a ověření správnosti údajů se data odešlou do databáze, po úspěšném přidání příspěvku vás na výsledek upozorní hláška, potvrzením vás stránka přesměruje na detailní stránku právě vytvořeného příspěvku.



Obrázek 3 - Přidat Auto

4.3 Editace příspěvku

Pro editaci příspěvku musíte znát heslo, které jste si založili při přidání příspěvku.

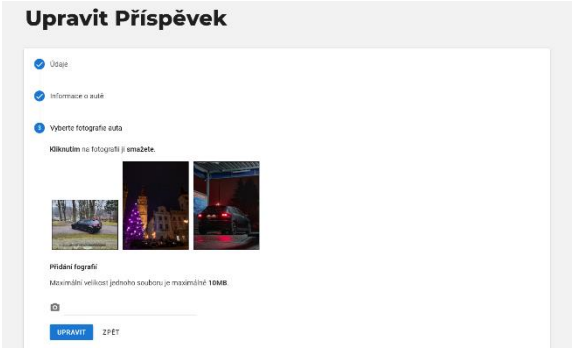
Po zadání správného hesla se vám následně zobrazí krokový formulář, ve kterém si můžete editovat hlavní informace příspěvku:

- Jméno majitele
- Informace o automobilu
- Fotografie automobilu

V posledním kroku, tj. editace fotografií se vám zobrazí náhled stávajících fotografií.

Kliknutím na náhled jednotlivé fotografie se fotografie smaže z příspěvku i z náhledu.

Přidání nových fotografií se provádí přes file input s ikonkou fotoaparátu stejně jako u přidání nového příspěvku. Nově přidaná fotografie musí splňovat určité požadavky o formátu a velikosti.



Obrázek 4 - Editace příspěvku

4.4 Sekce – Lidem se líbí

V této sekci se zobrazují příspěvky, které jsou nejlépe hodnocené veřejností. Pod touto sekcí můžeme naleznout „CTA – Call To Action“ tlačítko, které vyzývá uživatele k přidání svého automobilu do digitální garáže.

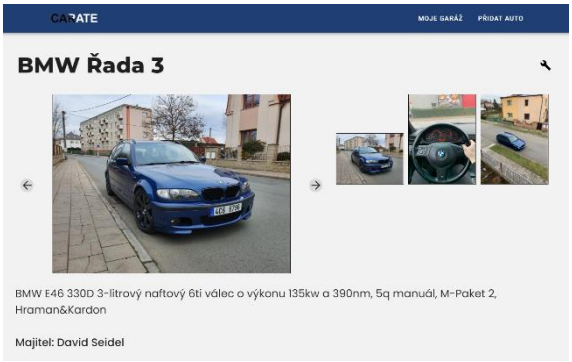
Obrázek 5 - Sekce Lidem se líbí

4.5 Detailní stránka příspěvku

Po rozkliknutí jakékoliv náhledové karty příspěvku se dostanete na detailní stránku příspěvku.

Zde můžete vidět hlavní údaje o automobilu:

- Značka a modelové označení
- Informace o automobilu
- Jméno majitele vozidla



Obrázek 6 - Detailní stránka příspěvku

Přehled fotografií je proveden následujícím způsobem. V levé části je náhled první fotografie, navigačními tlačítky s ikonou směrové šipky se dá listovat mezi ostatními fotografiemi. Druhý způsob listování je ukazatelem myši, náhledová fotka bude taková, na které byl ukazatel myši naposledy.

V pravém horním rohu je tlačítko pro editaci příspěvku, které vás přesměruje na editační stránku.

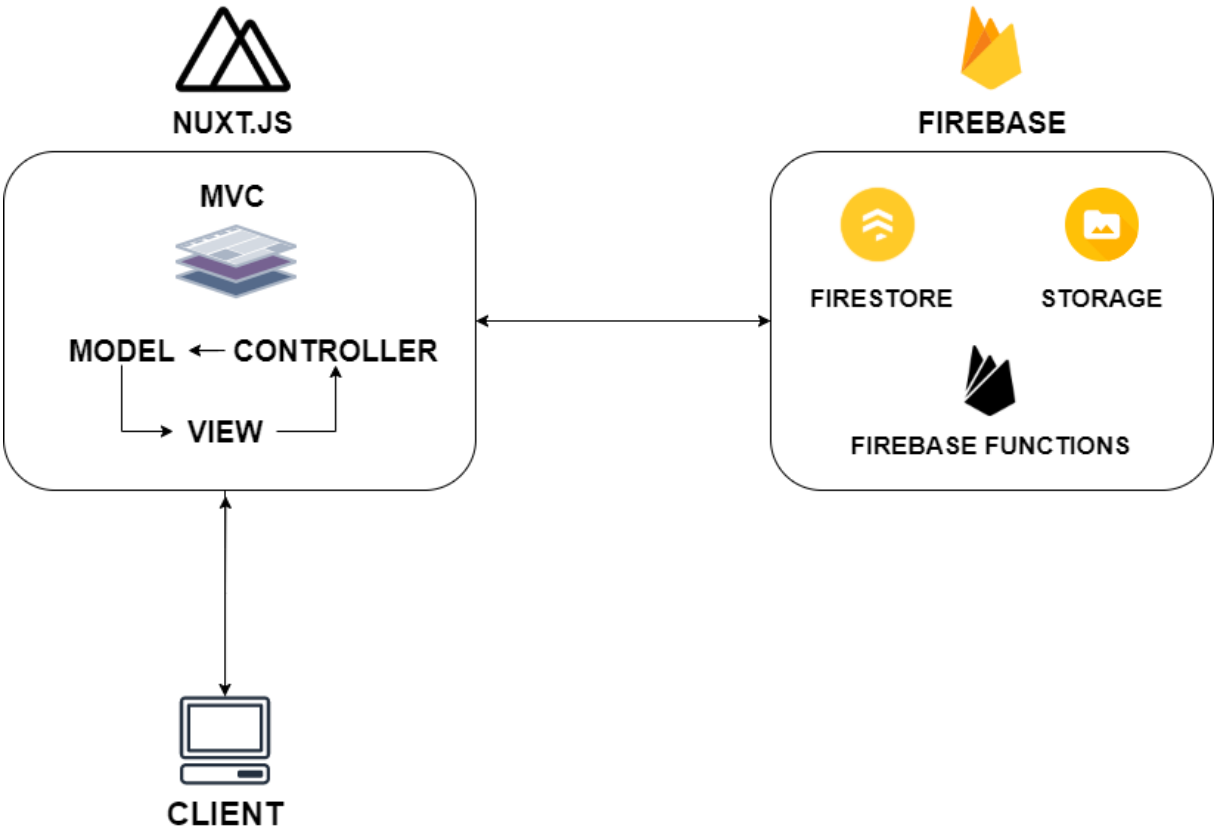
5 Systém

5.1 Architektura

Systém a jeho architektura je navržena co nejjednodušeji tak, aby byla aplikace rychlá a efektivní.

Klient se připojí na stránku a komunikuje s aplikací pomocí kontroléru, který zpracuje požadavek v modelu a výsledek se pošle do view, který se zobrazí uživateli na obrazovce. Kontrolér má přístup do back-end části, kterou tvoří kompletně technologie od společnosti Google – Firebase.

Zjednodušený model architektury můžete vidět zde:



Obrázek 7 - Architektura systému

5.2 Databáze

Webová aplikace používá databázi ve Firebase od společnosti Google, díky jejímu jednoduchému API a používání dokážeme docílit vysoké rychlosti a efektivity.

Firebase také nabízí přehlednou, lehce ovladatelnou webovou verzi, ve které můžeme spravovat veškeré záznamy a data.

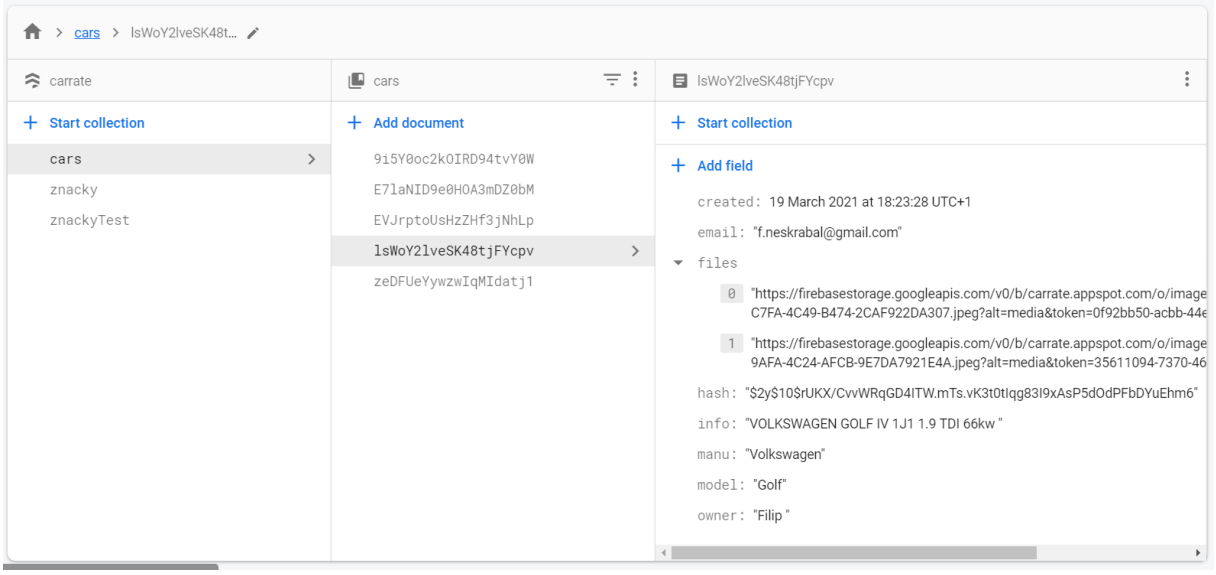
5.2.1 Cloud Firestore

Databáze Cloud Firestore je typ NoSQL databáze, která dokáže ukládat všechny základní datové typy, jako jsou: Stringy, Number, Boolean, Map, Array, Timestamp, Geopoint. Vše se v této databázi ukládá jako jeden velký JSON soubor.

V této aplikaci jsou 2 kolekce:

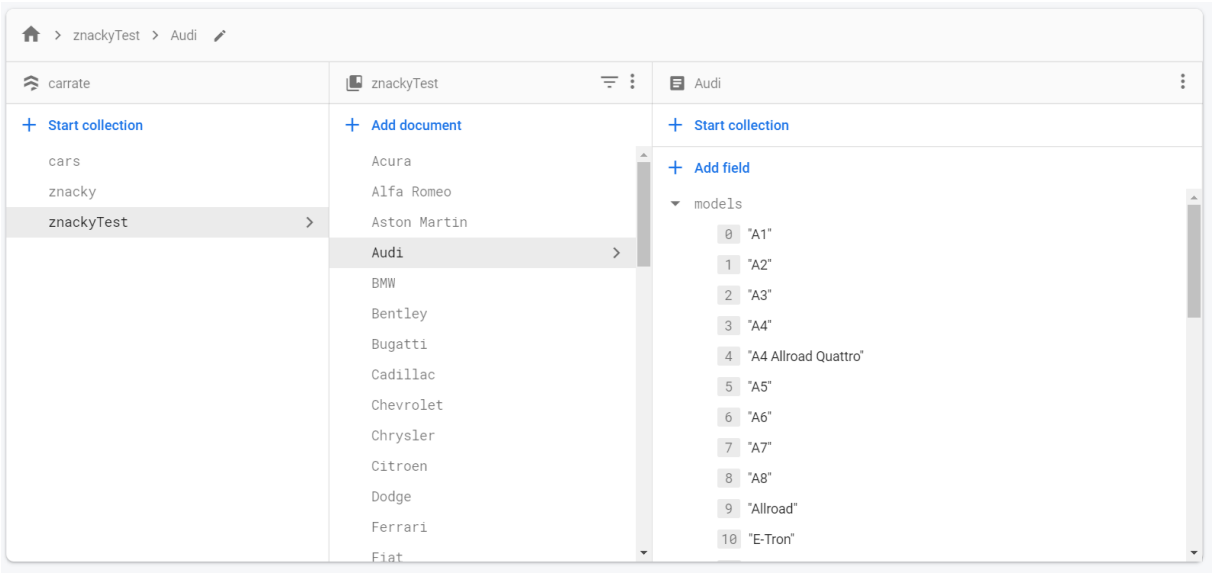
Kolekce „cars“, do které se ukládá každý příspěvek. Každý záznam nese své unikátní ID, pod kterým se dá vyhledat a má následující strukturu:

- **created** (Timestamp) – Datum a čas vytvoření příspěvku
- **email** (String) – Email připojený k příspěvku
- **files** (Array of Strings) – Pole odkazů na fotografie
- **hash** (String) – Hash hesla příspěvku
- **info** (String) – Informace o automobilu
- **manu** (String) – Značka automobilu
- **model** (String) – Modelové označení automobilu
- **owner** (String) – Jméno majitele



Obrázek 8 - Struktura Firebase, Záznamy

Kolekce „znacky“, do které bylo pomocí vlastního API vytvořená struktura, ve které jsou uloženy všechny značky a modely v následující struktuře:



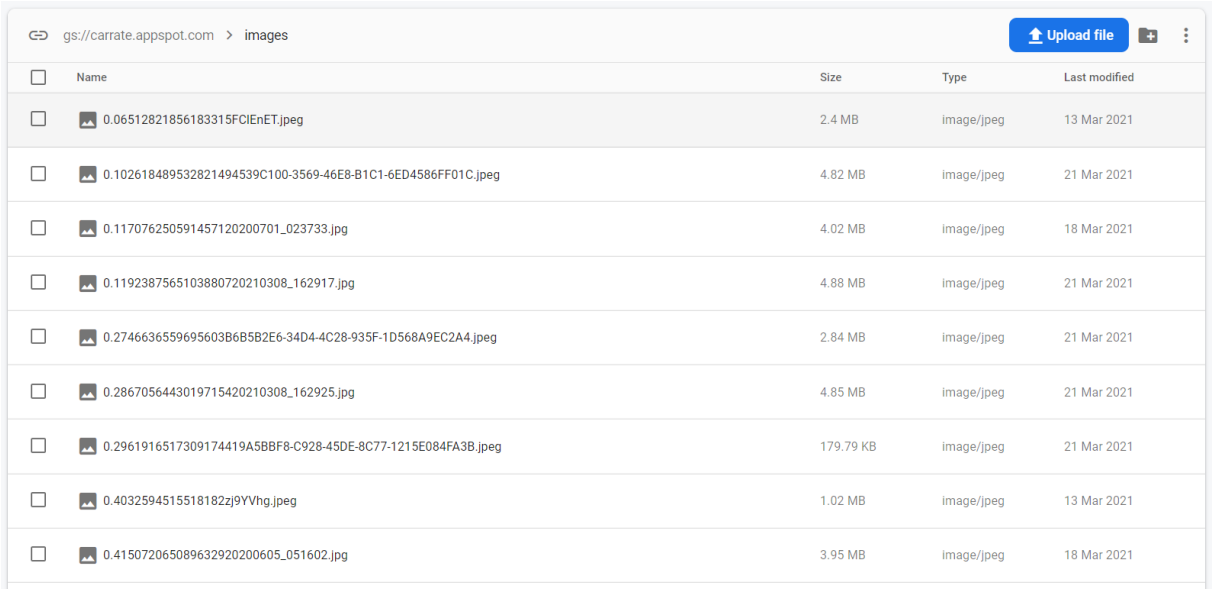
Obrázek 9 - Struktura Firebase, Značky

5.2.2 Firebase Storage

Tato technologie je tu použita pro ukládání obrázků do databáze.

Firebase poskytuje jednoduché API pro tuto technologii. Pomocí File inputu a funkce firebase.storage().ref().put(„soubor“) se dokáže uložit obrázek do uložiště Firebase Storage a vrátit dostuponou URL pro stažení či zobrazení.

V této webové aplikaci se duplicita názvu souboru řeší přidáním náhodného čísla před název souboru. Aplikace dokáže příjmat soubory ve formátech: jpg, jpeg, gif, tiff, webp, png a další.



Obrázek 10 - Firebase Storage

5.3 Hashování hesel

Ukládání hesel do databáze je zabezpečeno díky balíčku bcrypt.js, který je kompatibilní v C++, Node.js a také v klasickém prohlížeči. Je vytvořen v JavaScriptu týmem DCodeIO.

Tento balíček se snaží chránit proti různým druhům útoků např.:

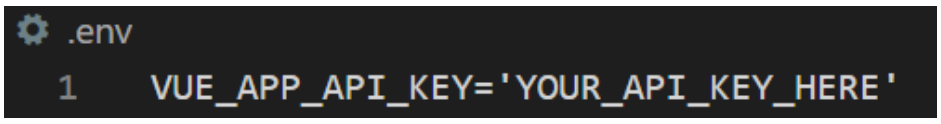
- **slovníkový útok**, kdy se předpokládá, že původní heslo je postaveno z nějakých existujících slov běžného jazyka, případně na konci doplněných o číslice apod. (4)
- **rainbow table útok**, jedná se o předem předpočítané tabulky dvojic „heslo – otisk“. Pro zkoumaný hash pak v tabulkách pouze stačí najít odpovídající původní heslo. Takové rozlomení je relativně rychlé. Rainbow tables jsou ale náročné na úložný prostor, proto se zpravidla opět počítají jen z omezené skupiny hesel, například slovníkových. (4)

V praxi jen použijeme funkci z tohoto balíčku, kam dáme jako vstupní parametr heslo, počet solících úrovní a o proces se to automaticky postará, dostaneme jako výsledek zahashovaný string, který uložíme do databáze. Máme zde i funkci, která porovnává zadané heslo s uloženým hashem.

5.4 Skrytí API klíče

Přístupový API klíč k Firebase produktům je potřeba si chránit. Neměl by být veřejně dostupný v kódu, ale ve skrytém souboru .env, který vytvoříme v root složce projektu. V projektu se jde poté API klíč získat pomocí kódu process.env.VUE_APP_API_KEY; (5)

API klíč nebude vidět ani v repozitáři např: GIT, protože v souboru .gitignore je už defaultně přidaný tento typ souboru.



Obrázek 11 - API Key – Hide

6 API

K celé této aplikaci byl vytvořený i „program/API“, které výrazně ulehčí práci vkládání všech značek a modelů do databáze.

Využívá se zde funkce Firebase Admin, která nám povolí měnit a upravovat strukturu a indexovat databázi.

Celý program spočívá v tom, že zdrojový soubor ve formátu JSON se přetvoří na požadovanou strukturu databáze a následně vyplní daty. Indexace databáze je potřeba kvůli následnému filtrování a vyhledávání v databázi Firestore.

```
const admin = require('./node_modules/firebase-admin');
const serviceAccount = require("./carrate-firebase-adminsdk-r7dkq-d0e9228527.json");
const data = require("./csvjson.json");
const collectionKey = "znackyTest"; //name of the collection
admin.initializeApp({
  credential: admin.credential.cert(serviceAccount),
  databaseURL: "https://carrate-default-rtdb.firebaseio.com"
});
const firestore = admin.firestore();
const settings = {timestampsInSnapshots: true};
firestore.settings(settings);

if (data && (typeof data === "object")) {
Object.keys(data).forEach(docKey => {
    console.log(data);
 firestore.collection(collectionKey).doc(docKey).set({models: data[docKey].models.split(",")}).then((res) => {
    console.log("Document " + docKey + " successfully written!");
}).catch((error) => {
   console.error("Error writing document: ", error);
});
});
}
```

Jsou zde 4 hlavní konstanty:

- **admin** – načtení Firebase Admin node modulu
- **serviceAccount** – konfigurace a klíč ke správě databáze (ke stažení v nastavení projektu Firebase)
- **data** – načtení dat z JSON souboru
- **collectionKey** – název cílové kolekce v databázi Firestore

7 Branding

Na začátku projektu byl jasně daný cíl vytvořit platformu, ke které bylo potřeba také vymyslet název a celkový motiv, kterým se bude nová platforma prezentovat na veřejnosti. Bylo tedy potřeba vytvořit název, logo a barevný motiv.

7.1 Název

Název CarRate vznikl kombinací slov „car“ (automobil) a „rate“ (hodnotit), vznikl poměrně rychle, protože jsou to klíčová slova této platformy.

7.2 Logo

Logo platformy CarRate vzniklo v online rastrovém grafickém editoru Photopea. (6)

První verze loga byla po dlouhém bádání nakonec ta finální. Slovo „Car“ končí písmenem „r“ a slovo „Rate“ začíná též písmenem „r“. Bylo tedy jasné, že klíčové bylo udělat grafické prolnutí písmena „R“. Bylo toho docíleno kontrastem černé a bílé barvy.

Obrázek 12 - Logo Platformy

8 Závěr

V těchto několika měsících se mi povedlo vytvořit základ této sociální sítě, provést alpha-testy pro optimalizaci aplikace. Webová verze je aktuálně připravena pro beta test a spuštění pro blízkou veřejnost.

Při práci na projektu jsem narazil na mnoho problémů, ať už s nějakými limity technologií či přístupností dat. Vše se ale s nějakým úsilím a pomocí ostatních vyřešilo.

Do aplikace je v plánu implementovat další funkce např.:

- Kalendář událostí (srazů, projížděk)
- Hodnocení automobilů
- Mobilní aplikace

Po spuštění projektu pro širokou veřejnost bude potřeba i propagace, tedy propagační materiál, reklamní spoty a tak dále.

9 Reference

1. ***Nuxt.js.*** **[Online] https://nuxtjs.org/.**

2. *Firebase.* [Online] https://firebase.google.com/.

3. *Vercel.* [Online] https://vercel.com/.

4. *Phpguru.* [Online] 2021. https://www.phpguru.cz/clanky/soleni-hesel.

5. *API Key Hide.* [Online] https://betterprogramming.pub/how-to-hide-your-api-keys-c2b952bc07e6.

6. *Photopea.* [Online] https://www.photopea.com/.

10 Seznam obrázků