**Tematika**: Razširitev API točke podatkov o uporabi javnih ekoles v občini Tržič 2023

**Avtor**: študent Samir Subašić SS7581

**Uvod**: V seminarski nalogi, bom sodeloval sam. Predstavil bom statistiko uporabe javnih koles v občini Tržič. Po prvotnih ad-hoc verzijah reševanja seminarske naloge, sem ugotovil, da bo najbolj zanimiva rešitev, razširitev API podatkovne točke o uporabi javnih ekoles v občini Tržič, za leto 2023.

Ogledal si uporabo obstoječe točke in strukturo podatkov, ki je na voljo na API podatkovni točki: [Swagger UI (musiclab.si)](https://trzic.musiclab.si/openapi/swagger#/default/gorenjskabike_gorenjskabike_get)

**Problematika**: Kako bi najbolje predstavil, končnemu uporabniku zajete podatke, ki jih pridobim iz API podatkovne točke. Namreč, klicanje podatkov o statistični uporabi ekoles, na obstoječi API točki, ni najbolje zastavljena z parametri **page** in **size**. Za boljše razumevanje končnim uporabnikom in za razvijalce različnih nivojev znanj, da se podatki zbirajo po datumih in urah. Kar pomeni, podamo začetni datum in uro in nato končno datum in uro. S podanim časovnim intervalom, pridobimo bolj natančno podatke in to je ljudem bolj razumljivo, kot spoznavati **page** in **size** parametre.

**Cilj predstavitve**: želim prikazati uporabne (relevantne) podatke uporabniku iz množice podatkov in parametrov. Podatki bodo lahko predstavljeni na spletni strani, javni projekciji (lcd zaslon v javnih prostorih), mobilnih napravah. Možno bo novo podatkovno API točko samostojno klicati preko http protokola in API točka bo posredovala json podatke. Tako bodo lahko razvijalci vključili, po svojih željah podatke iz točke.

**Pridobitev podatkov**: podatke pridobim iz javne API točke (musiclab.si). Prvi korak je raziskovanje podatkovne točke in sem opazil, da so vsi podatki zbrani v uporabi parametrov **page** od 1 do 1015 v velikosti size je 1000 (1000 vrstic za 1 page). Za uporabo svoje točke, upoštevam, da je sezona za letos končana in tako ne bo več novih podatkov. V prvem koraku moja API točka pretoči vse podatke iz prvotne API točke (/api/gorenjskabike) in jih shrani lokalno v datoteko »allData.json«. Tako so podatki spravljeni v lokalni cache. Datoteka je velikosti preko 300MB in vsebuje preko 1 miljon vrstic. Podatke streže moja API točka, ki je napisana v javascriptu in jo poganja node.js okolje. Sestavljena je iz proxy.js in server.js

**Programsko pridobivanje:** da bi se izognil težavi **CORS** (Cross-Origin Resource Sharing), bom uporabil node.js (express) komponento, ki mi bo omogočala programsko pridobivanje podaktov,  
( <https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS> )

Podatke lahko dobivam direktno iz točke /api/gorenjskabike ali shranjene v lokalno kopijo, ki so pretočene iz omenjene točke. Lahko nastavimo periodičen mehanizem, glede na operacijski sistem, ki periodično zbriše po nastavljenem časovnem intervalu »allData.json« lokalno datoteko, kar privede, da mora moja aplikacija pretočiti nove ažurirane podatke na lokalni disk. Tako so podatki na voljo za nadaljnjo podatkovno analizo z mojim namenskim web lokalnim orodjem ali s direktnim klicem http na mojo točko.

**Inštalacija:** Zadevščino inštaliram tako, da naprej inštaliram node.js in nastavim klicanje »node« ukaza v $PATH. Nato inštiram node.js komponente v terminalu ali konzoli (ali v Command Promptu):  
**npm install express node-fetch fs**

**Izvajanje:** Zaženemo v konzoli/terminalu/Command Prompt:  
node server.js  
  
Sedaj imamo pripravljeno, da izvajamo klice preko localhost originalnega API-ja:  
<http://localhost:3000/api/gorenjskabike?page=1&size=1000>

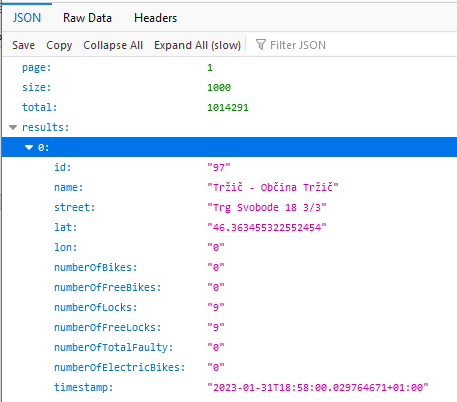
**Parametri podatkov:** sta **page** in **size**

**Podatki za leto 2023, razdeljeni so na parametra page in size.** Prvi predstavlja strani, drugi predstavlja velikost te strani. Največja velikost ene strani je lahko 1000. Pri pregledovanju podatkov sem ugotovil, da je pri velikosti strani 1000, število vseh strani je 1015. Vse vnosov za leto 2023 je 1014291 ali milijon vnosov za posamezno postajo.  
  
Novi naslov je:  
<http://localhost:3001/api/localdata?selectedDateTimeFrom=2023-04-19T21:00&selectedDateTimeTo=2023-04-19T21:20>

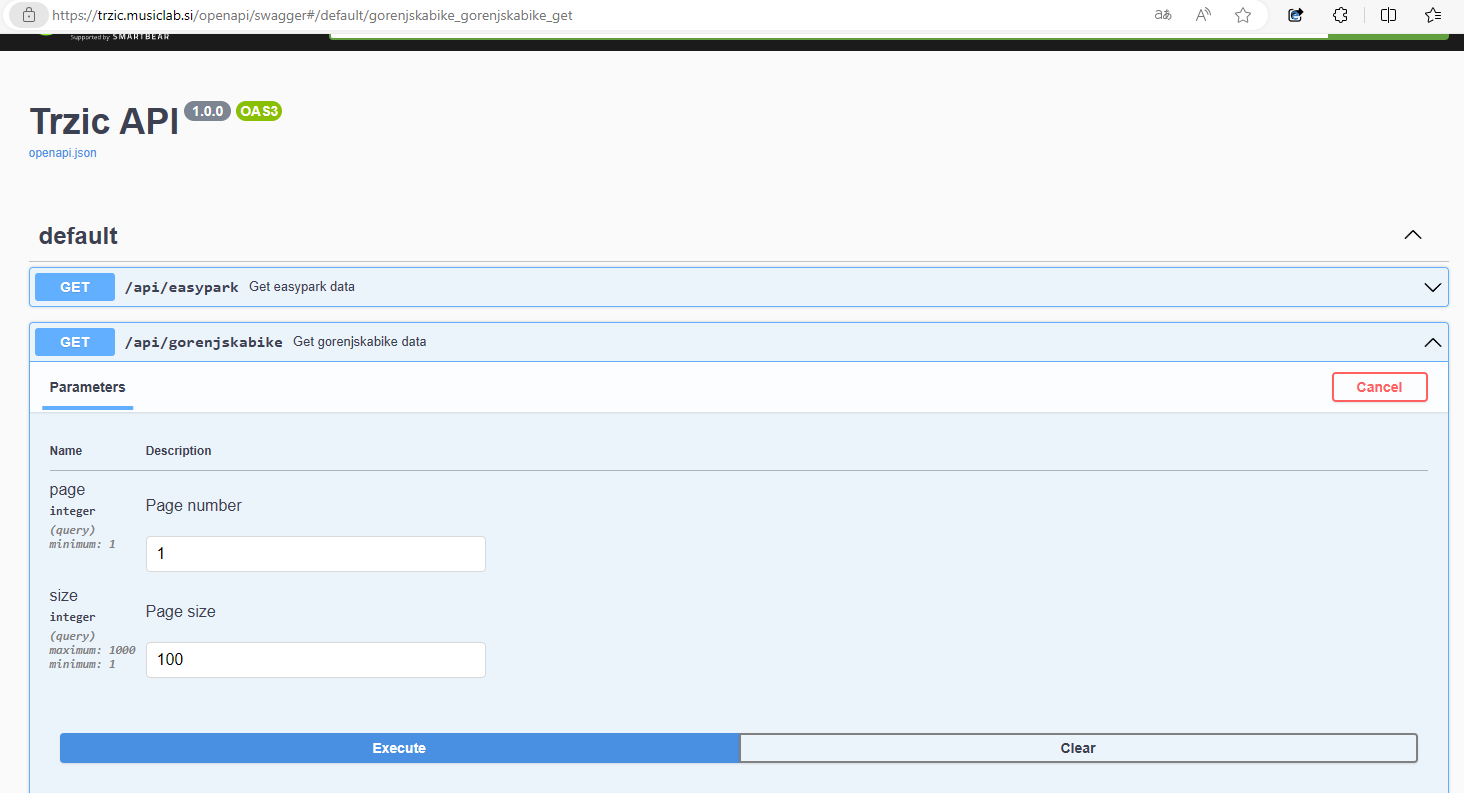
**Parametri podatkov: selectedDateTimeFrom** in **selectedDateTimeFrom  
Časovni format: yyy-mm-ddThh:mm,** npr.: 2023-11-18T21:21

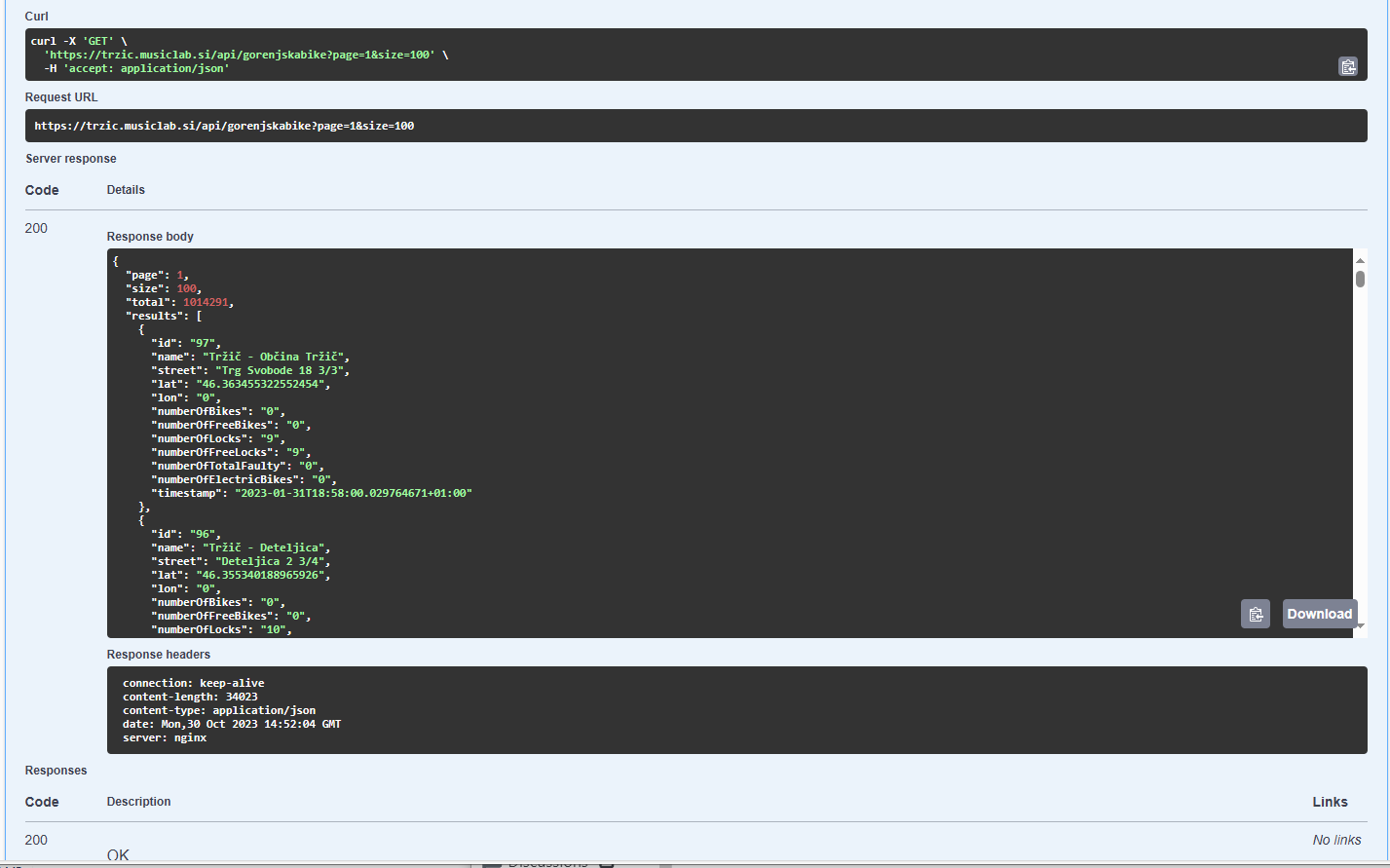
**Podatki se obdelujejo v server.js, ker je hitrejše v node.js, kakor prenašat celotno vsebino allData.json in obdelati lokalno v spletnem brskalniku. Server pošlje le obdelane podatke v json obliki.**

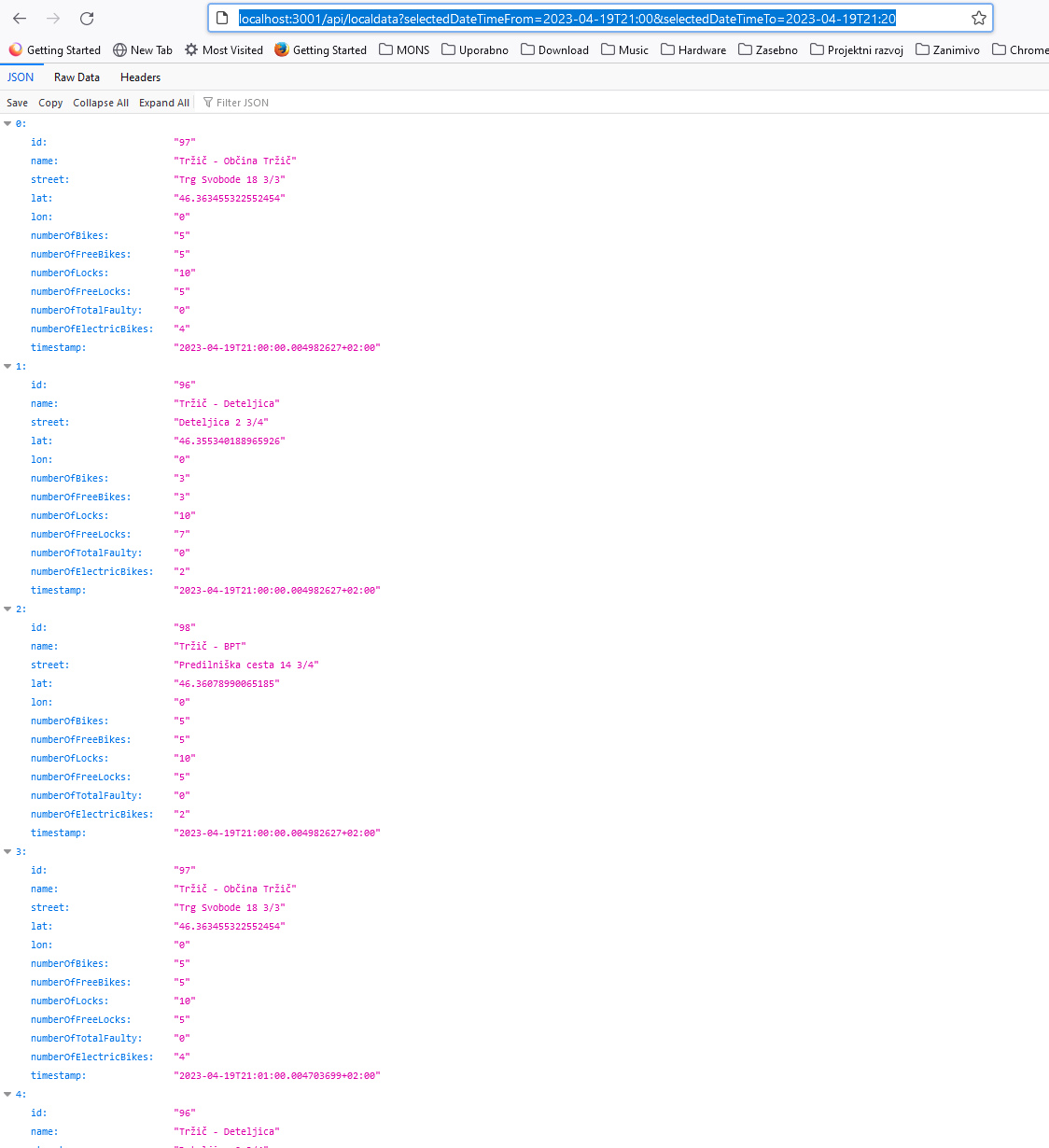
**Slika1**: Odgovor API točke, glede »gorenjskabike« podatkov



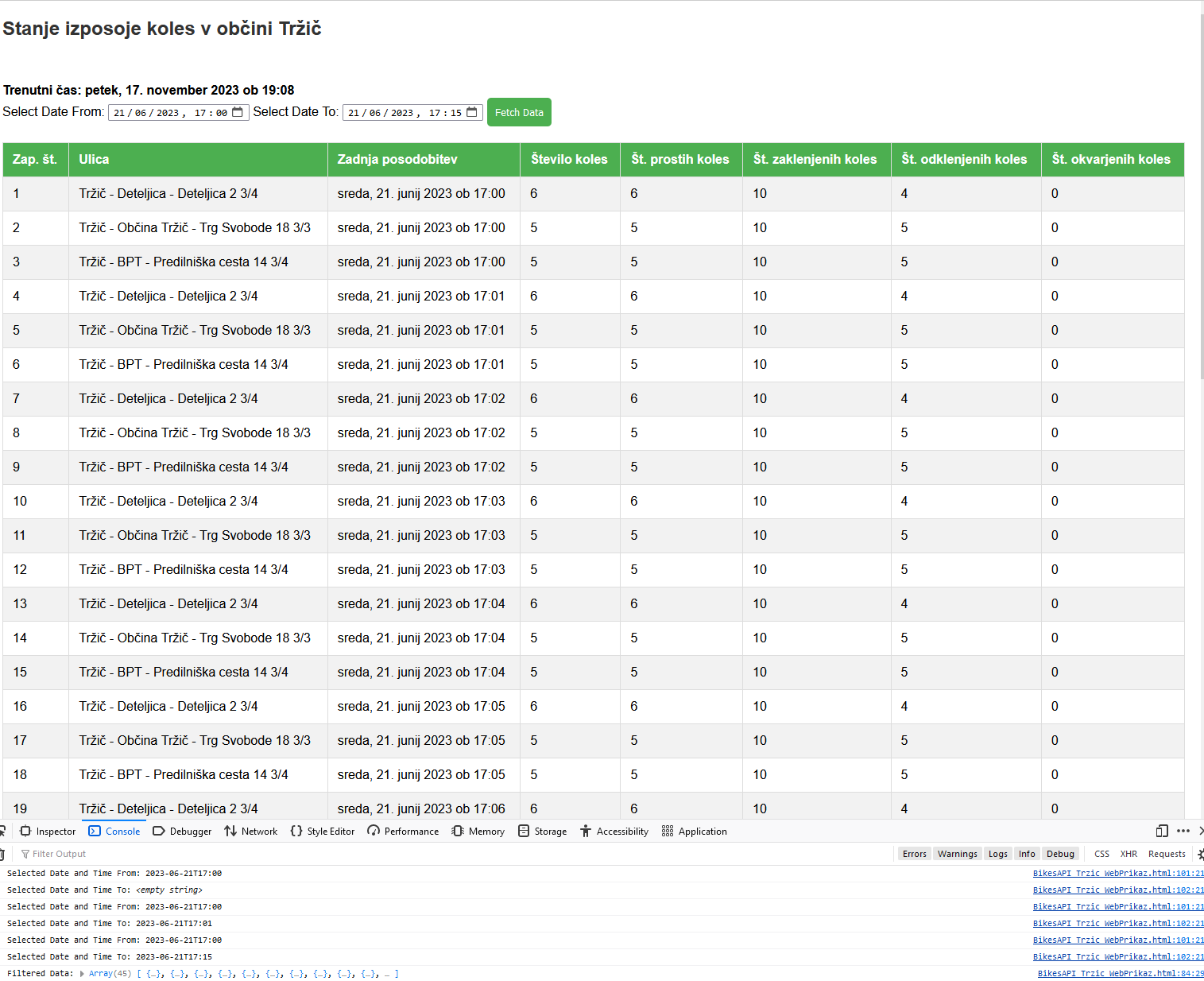
**Slika2**: Trzic API 1.0.0, kjer si ogledam podatke

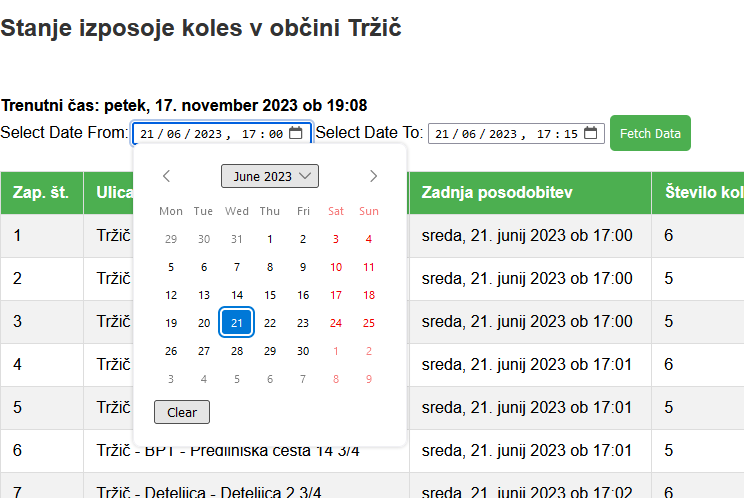


**Slika3:** prikaz JSON podatkov po posredovani zahtevi na API točko:  


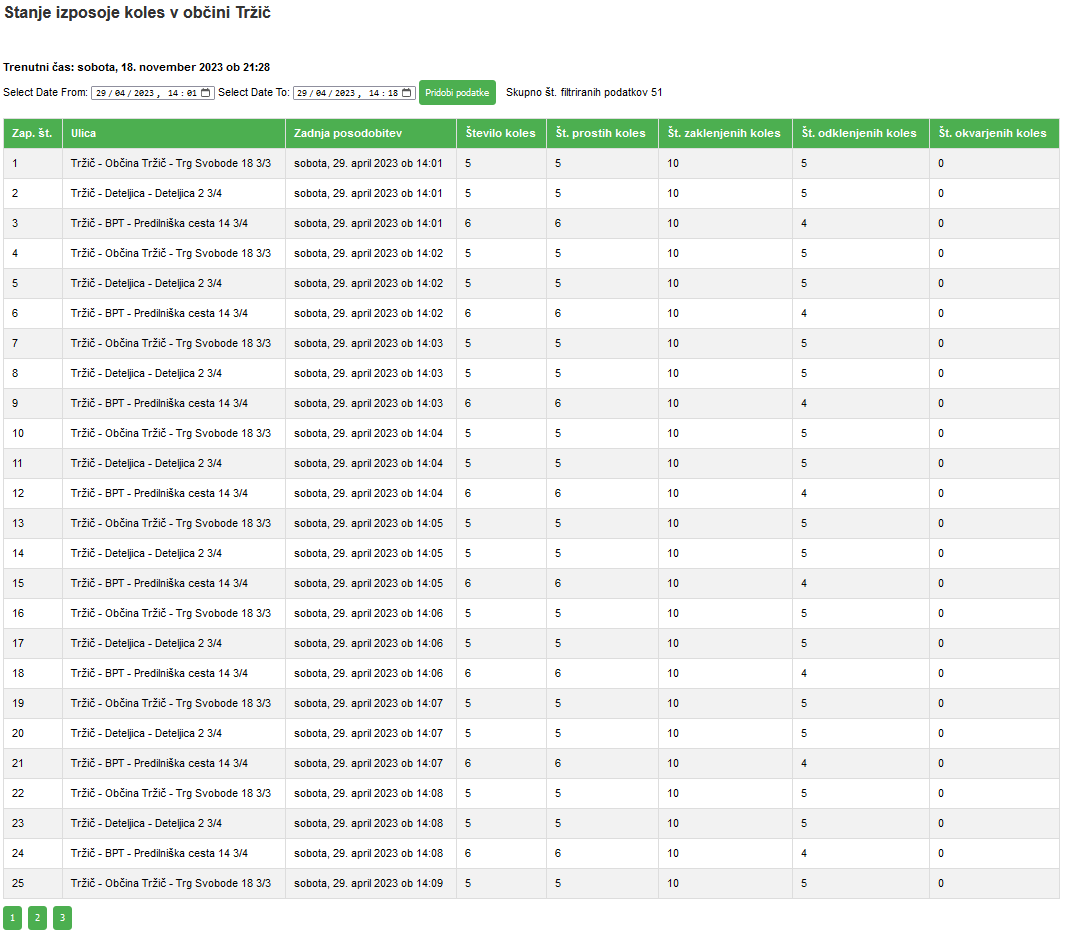
**Slika4:** direktni API klic na mojo lokalno točko ****

**Slika5:** Prva verzija lokalne web strani

****

**Slika6:** Dodani interaktivni komponenti, za določanje začetnega in končnega datuma in ure  
****

**Slika7:** Končna interaktivna verzija, z oštevilčenjem strani, prikazom št. vrnjenih podatkov in itd.

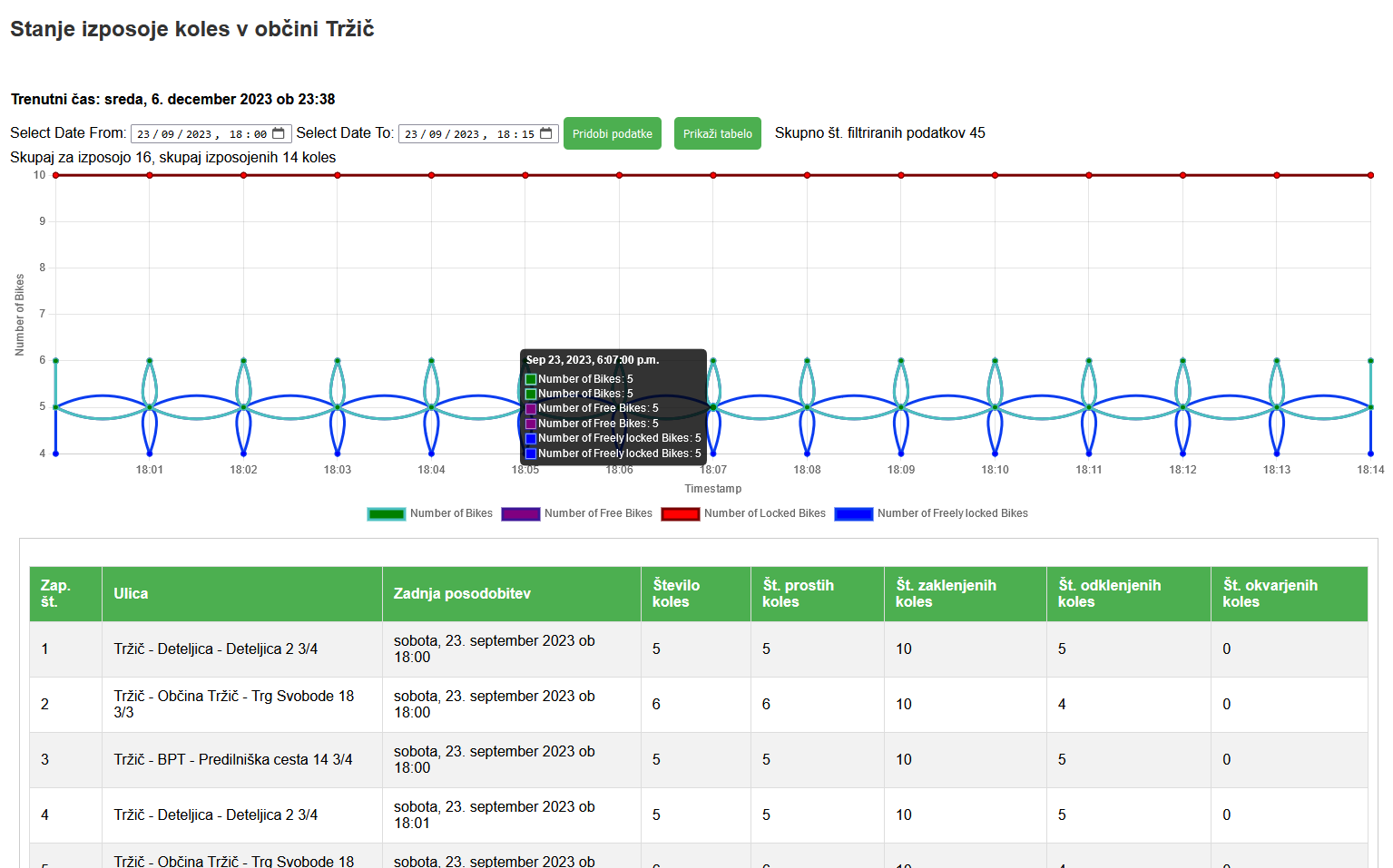


**Končna verzija lokalne web strani:**   
Vsebuje interaktivne dinamične html komponente. Nisem uporabil kompliciranih javascript knjižnic, ker bi rad omogočil, da lahko razvijalci po svoje prirejajo prikaz rezultatov. Uporabljen je le čisti javascript. Spletna stran je dinamično zasnovana, tabela se dinamično izrisuje, glede na posredovanje zahteve pri pritisku na gumb »Pridobi podatke«. Dodana je osnovna validacija podatkov, če manjkajo vneseni datumi. Dodano je dinamično oštevilčenje tabel. Maksimalno se prikaže 25 zapiskov na stran. Zaraven gumba »Pridobi podatke« je zapisano, koliko podatkov je izluščil node.js server.  
  
Uporabniku sistema izposoje ekoles predstavim:  
- na kateri postaji in lokacija izposojevalne postaje  
- zadnja posodobitev ažurnosti podatkov  
- število koles in število prostih koles sta enaka podatka  
- število prostih koles, ki so na voljo za izposojo  
- število zaklenjenih koles  
- število odklenjenih koles  
- število okvarjenih koles

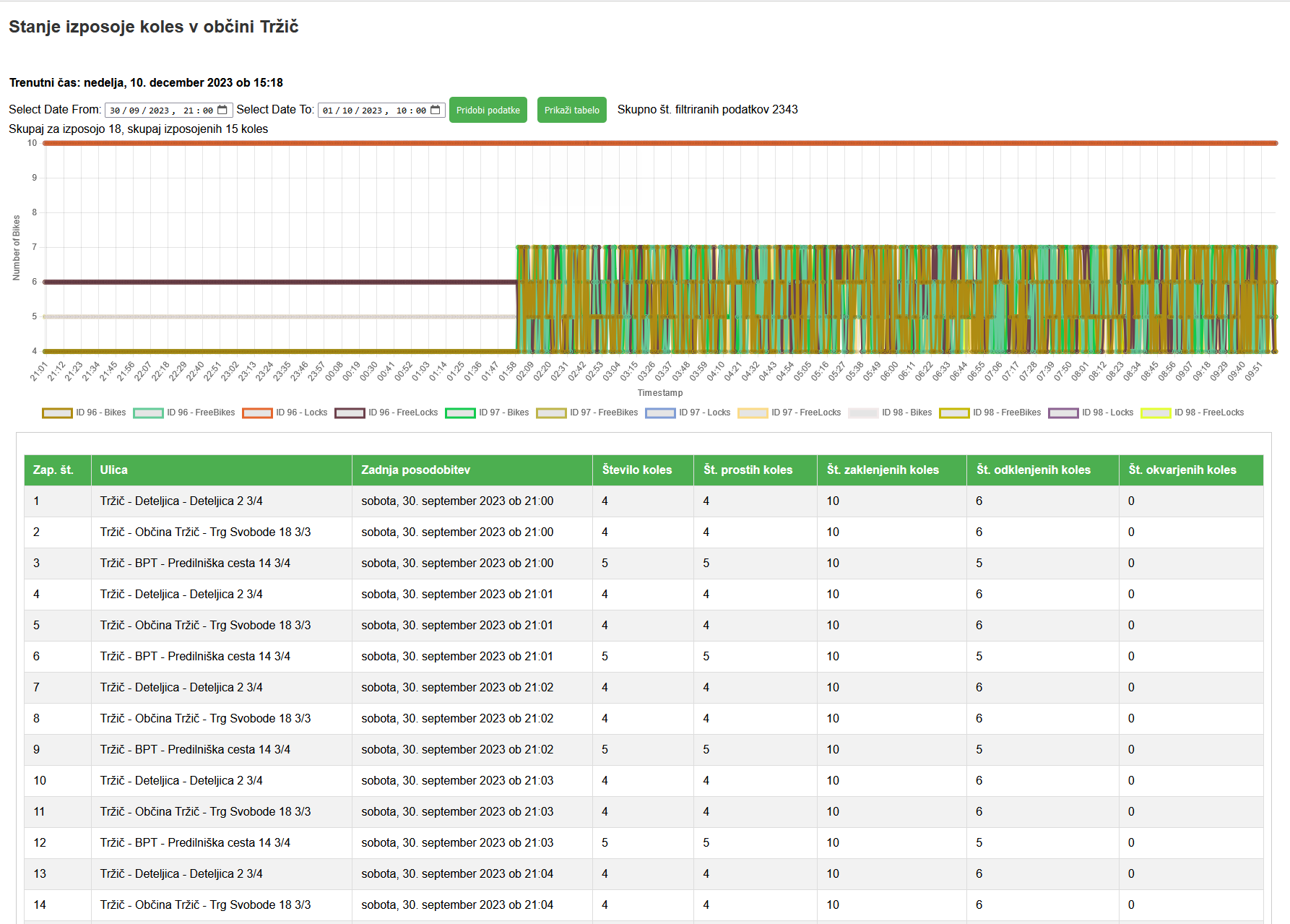
**Zaključna točka in končna misel:** želja je, da se uporabnikom v občini Tržič predstavijo statistični podatki na preprost način z (iskanje po datumih in) uporabo tabele, ki so jo lahko programsko uporabi ali integrira naprej v različnih prikazovalnikih, internetna stran, mobilna aplikacija, javni plakat zaslon in podobno.

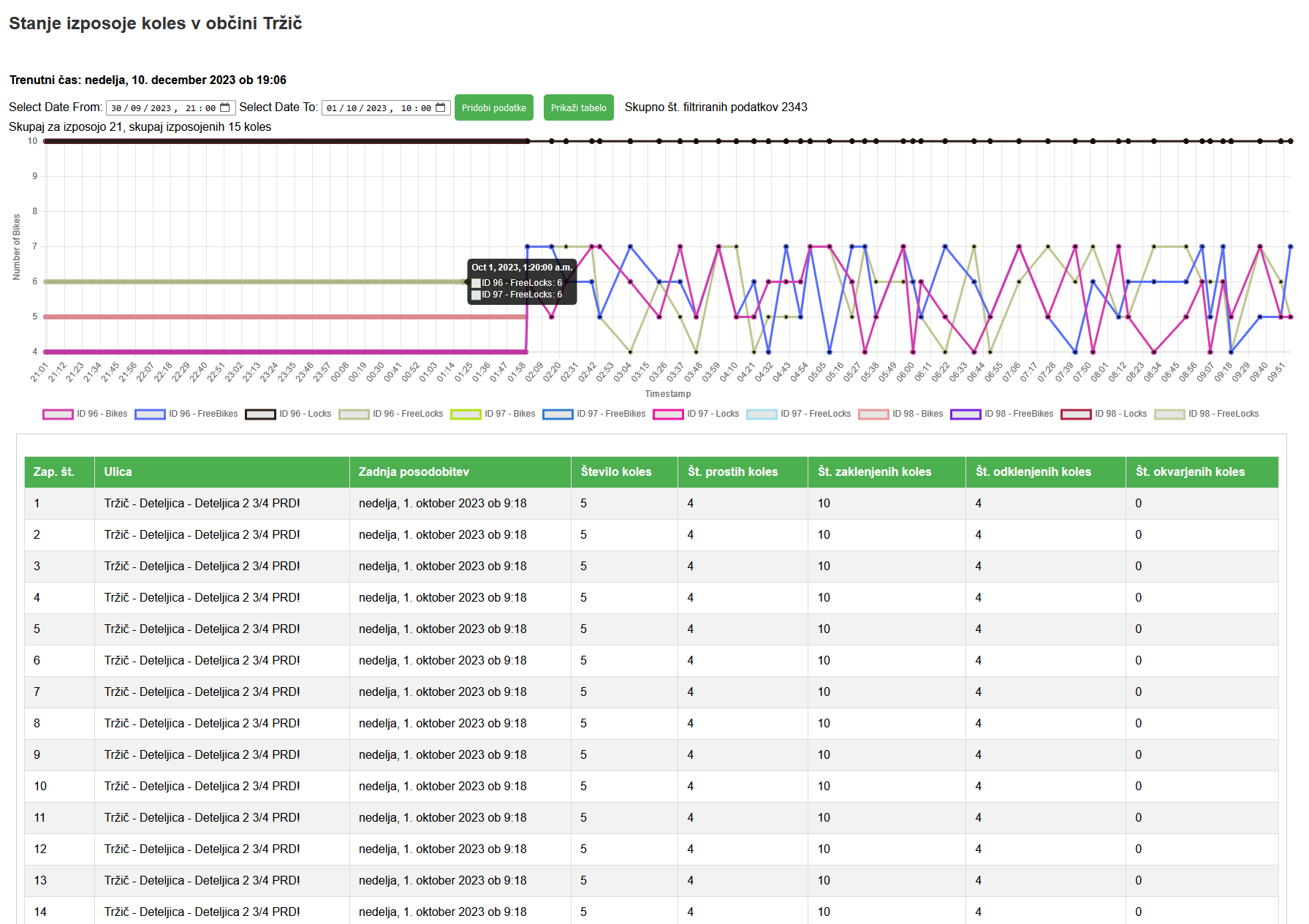
**Končno delo in izvorna koda:** je na voljo na githubu repozitoriju  
<https://github.com/SammieLJ/gorenjskabike_api>

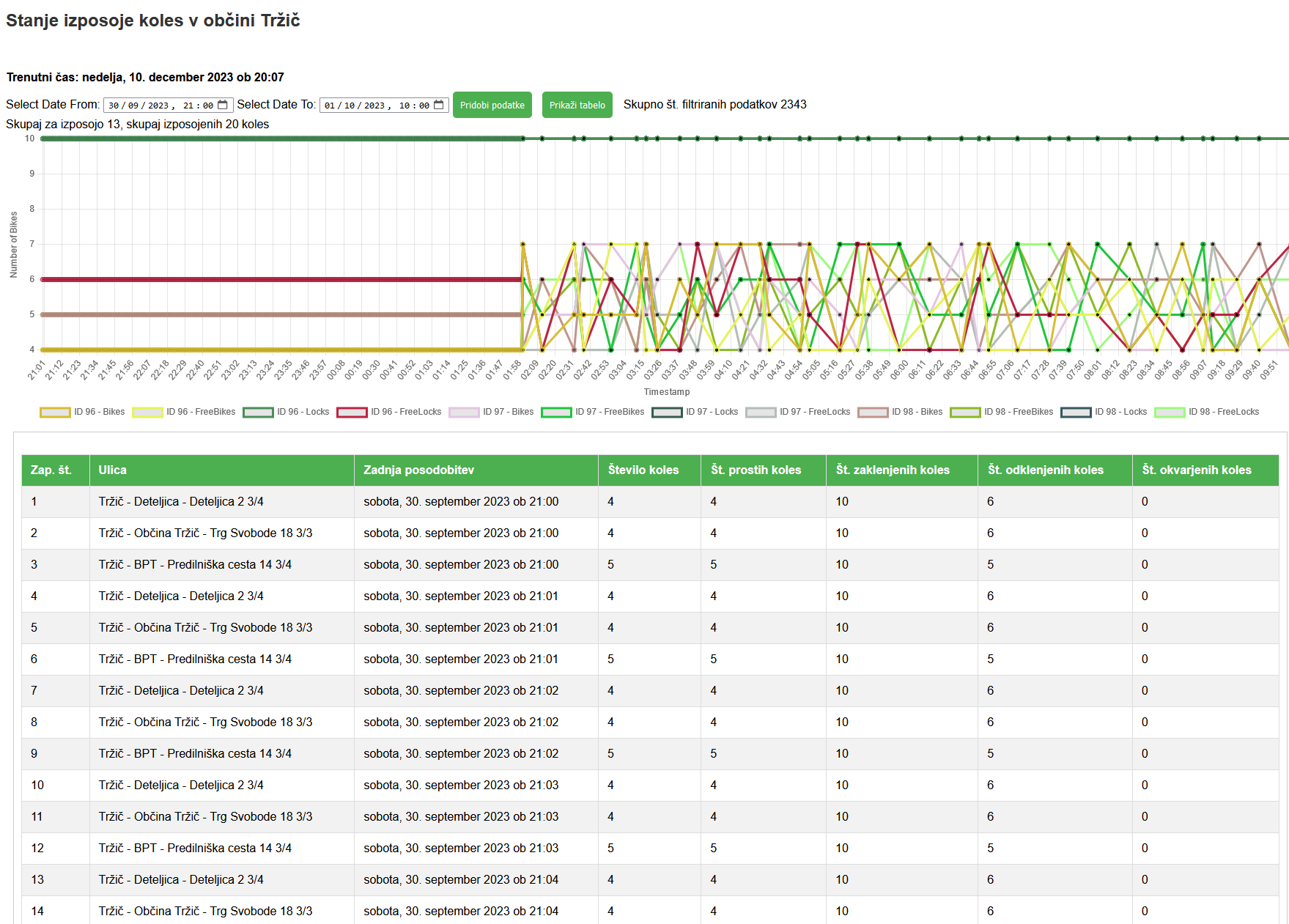
**Posodobitev:**











Povečanje povprečnih voženj od 10 do 40 min  
