# Tehnična dokumentacija projekta rt21

**David Slatinek** **Marcel Iskrač** **Vid Kreča**

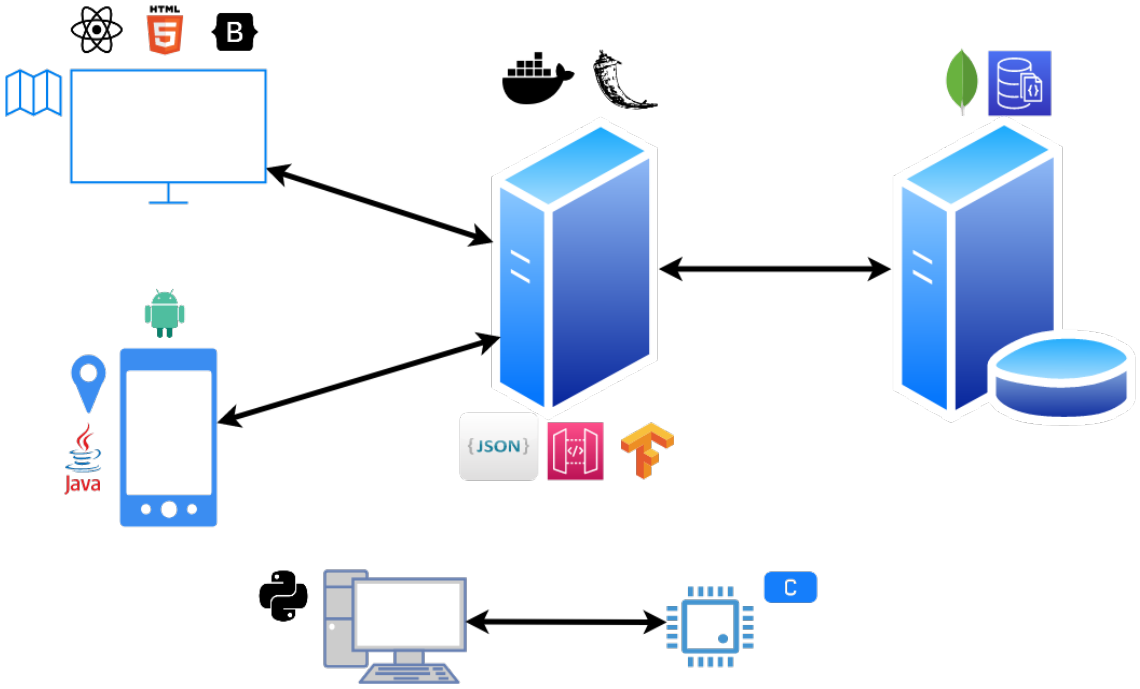
Maribor, december 2021

**Povzetek**

Ta dokument predstavi tehnično dokumentacijo projekta rt21. Najprej predstavimo projekt kot celoto in njegove glavne enote. Na koncu predstavimo posamezno enoto, med katere spadajo podatkovna baza, spletni vmesnik za pridobivanje in shranjevanje podatkov z računalniškim vidom, Android aplikacija, spletna stran, algoritem za stiskanje zaporedja števil in uporaba mikrokrmilnika.

## 1 Enote projekta

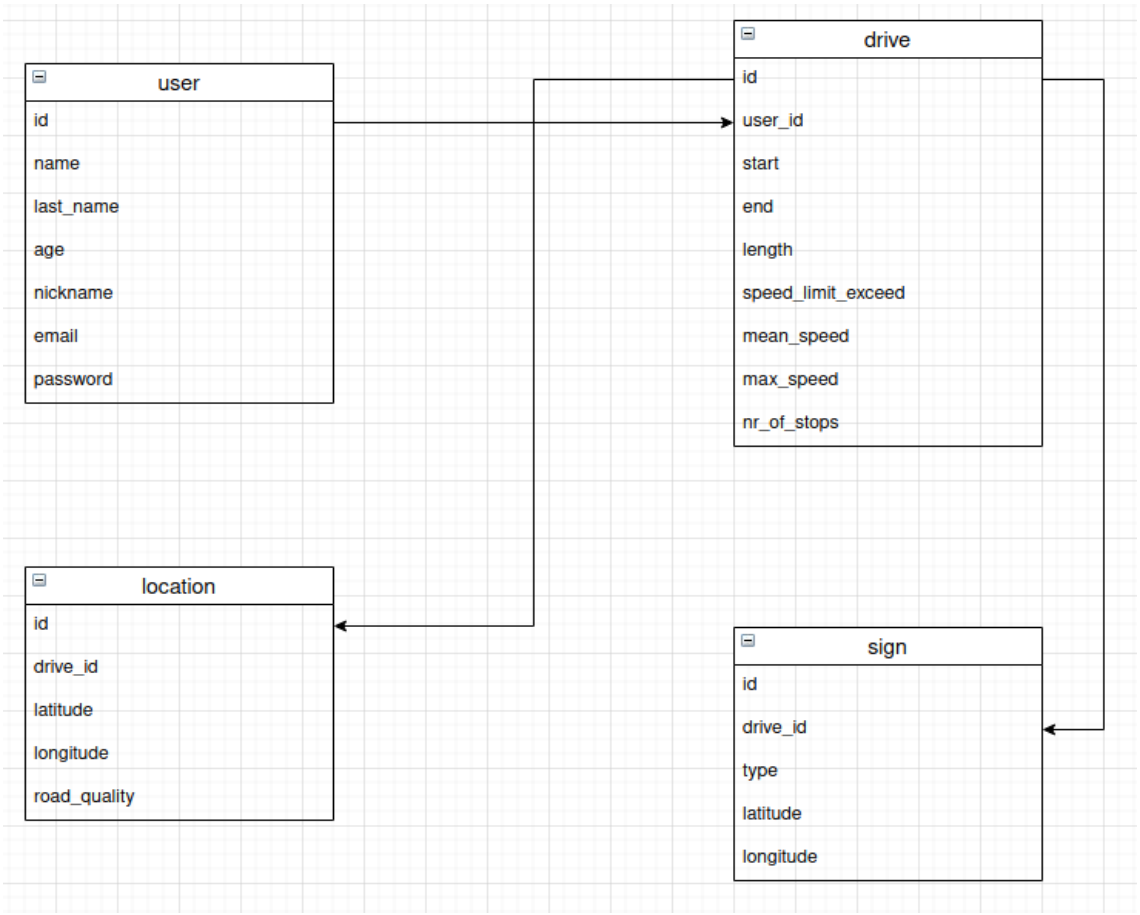
Projekt sestavlja 6 glavnih komponent: podatkovna baza, spletni vmesnik z računalniškim vidom, Android aplikacija, spletna stran, algoritem za stiskanje podatkov in mikrokrmilnik - povezave med enotami in uporabljene tehnologije prikazuje slika 1.




Slika 1: Enote projekta.

### 1.1 Podatkovna baza

Za podatkovno bazo smo izbrali NoSQL tip baze, natančneje MongoDB. Za gostovanje podatkovne baze smo uporabili MongoDB Atlas, ki ima različne nivoje gostovanj, izbrali smo zastonj verzijo. V bazi shranjujemo podatke o uporabniku, njegovih vožnjah, lokacijah teh voženj in podatke o prometnih znakih [1]. Slika 2 prikazuje ER diagram, ki smo ga naredili s spletno aplikacijo Diagrams.net.




Slika 2: ER diagram.

## 1.2 Spletni vmesnik z računalniškim vidom

API je narejen s programskim jezikom python oziroma z njegovim ogrodjem, flask in temelji na arhitekturi REST, ki vsebuje HTTP metode [1]. API gostuje na platformi Heroku in deluje znotraj docker zabojnika. Za hitrejše delovanje smo uporabili strežnik *gunicorn*. Podatki se vračajo v obliki JSON. Kar se dotika varnosti, sistem vsebuje naslednje varnostne mehanizme [1]:

- Identifikacija - API ključ.
- Šifriranje prometa - TLS[1].

Podpiramo vse operacije CRUD:

- C - Create - ustvarjanje novih zapisov.
- R - Read - pridobivanje podatkov.
- U - Update - posodabljanje podatkov.
- D - Delete - brisanje podatkov.

Za prepoznavo prometnih znakov iz slike smo uporabili knjižnice TensorFlow, Keras in OpenCV - z njimi smo ustvarili konvolucijsko nevronsko mrežo. Ob prejemu slike API prepozna prometni znak iz slike, nato pa to vrednost pošlje nazaj k odjemalcu. Za shranjevanje gesla kot hash smo uporabili knjižnico Flask-Bcrypt,

[1]Transport Layer Security; poskrbi Heroku.

za povezovanje na bazo pa knjižnico Flask-PyMongo. Prav tako smo - zaradi avtomatizacije - napisali bash skripto, ki zgradi docker sliko in jo naloži na strežnik. API smo razvili z razvojnim okoljem PyCharm Professional, za testiranje delovanja smo uporabili program Postman.

## 1.3 Android aplikacija

Android aplikacijo smo razvili v razvojnem okolju Android Studio ter s programskim jezikom Java. Aplikacija omogoča prijavo in spremembo gesla uporabniškega računa. Glavna funkcionalnost aplikacije je zajem slike in podatkov iz senzorjev[2] ter pošiljanje teh na strežnik [1]. Dodatno smo uporabili knjižnico OpenStreetMap za prikaz trenutne lokacije.
V aplikaciji smo uporabili naslednje senzorje:

- Kamera - zajemanje slike za prepoznavo prometnih znakov.
- GPS - dostop do zemljepisne širine in dolžine.
- Žiroskop - kakovost ceste.
- Pospeškometer - spremljanje hitrosti.

## 1.4 Spletna stran

Spletno stran smo ustvarili s pomočjo JavaScript knjižnice React, HTML5 in CSS ter Boostrap knjižnice za enostavno oblikovanje elementov. React smo uporabili za postavitev in klic komponent aplikacije in za komunikacijo med API-jem in spletno stranjo [1]. Dodatno smo uporabili knjižnico Leaflet za prikaz zemljevida in podatkov na njem. Glavna funkcionalnost spletne strani je vizualizacija podatkov, poleg tega aplikacija omogoča še registracijo, prijavo in dostop do profilne strani.

Spletna stran gostuje na platformi Heroku, znotraj docker zabojnika in podpira HTTPS protokol. Za hitrejše delovanje smo uporabili strežnik Nginx. Tudi tukaj smo - zaradi avtomatizacije - napisali bash skripto, ki zgradi docker sliko in jo naloži na strežnik. Spletno stran smo razvili z razvojnim okoljem WebStorm, PhpStorm in urejevalnikom Visual Studio Code.

## 1.5 Algoritem za stiskanje zaporedja števil

Algoritem za stiskanje zaporedja števil smo razvili s programskim jezikom python, v razvojnem okolju PyCharm Professional. Za testiranje smo uporabili program Postman. Algoritem je uporabljen v APIju. Ko spletni vmesnik dobi zahtevo, se izvedejo naslednje operacije:

- Pridobijo se vrednosti kakovosti ceste iz podatkovne baze.
- Števila se zapišejo v datoteko[3].
- Izvede se stiskanje podatkov.
- Binaren zapis stisnjenih podatkov v datoteko.
- Pošiljanje datoteke odjemalcu.

Odjemalec lahko po prejemu datoteke izvede dekompresijo in uporabi števila v nadaljnji obdelavi.

## 1.6 Uporaba mikrokrmilnika

Na mikrokrmilniku STM32F3DISCOVERY smo uporabili pospeškometer, s katerim na nizkonivojskem nivoju pridobimo podatek iz senzorja. Program za mikrokrmilnik smo razvili s programskim jezikom C, v razvojnem okolju STM32CubeIDE. Nato smo s pythonom v PyCharmu izdelali konzolno aplikacijo, ki pridobi ta podatek iz mikrokrmilnika preko USB povezave.

[2] Vsakih 5 sekund.
[3] Algoritem deluje na način branja števil iz datoteke.

### 1.7 Ostalo

Za verzioniranje projekta smo uporabili git, celoten projekt pa je tudi na voljo na githubu. Za komunikacijo smo uporabljali discord. Dokumente smo naredili z LaTeXom, na spletnem urejevalniku Overleaf.

## Literatura

[1] David Slatinek in Marcel Iskrač. "Projekt rt21". Jun. 2021. (Pridobljeno 21. 12. 2021).