# Funkcijska aplikacija za obradu podataka u stvarnom vremenu


*Izradili:*

*Laura Lončarić*
*Matej Višnjić*

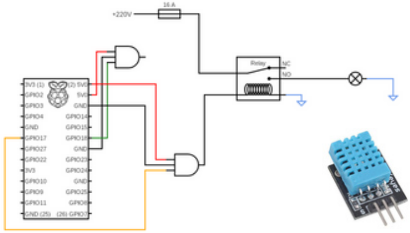
*Sveučilište Jurja Dobrile u Puli*
*Fakultet Informatike u Puli*
*Funckijsko programiranje*


*Tema ovog projekta je izgradnja funkcionalne aplikacije za obradu podataka u stvarnom vremenu, s fokusom na prikupljanje podataka o okolišu tijekom duljeg razdoblja i provođenje statističke analize.*

## 01. Arhitektura sustava i dizajn procesora za streaming

Glavni cilj je učinkovito upravljanje velikim tokovima podataka i njihovo prikazivanje na intuitivan način. Sustav koristi Raspberry Pi za prikupljanje podataka i Haskell za obradu i analizu.

- **DHT11 senzor:** Mjeri temperaturu i vlagu svakih 30 minuta.
- **Raspberry Pi:** Povezuje se sa senzorom putem GPIO pinova i prikuplja podatke pomoću Python skripte.
- **Pohrana Podataka:** Prikupljeni podaci se spremaju u logove sa vremenskim oznakama, temperaturom i vlagom.
- **Obrada u Haskellu:** transformacija i filtriranje podataka, agregacija podataka za izračun statističkih pokazatelja
- **Grafički Prikazi:** Generiranje grafikona i histograma za vizualizaciju promjena u temperaturi i vlagi.
- **Flask API:** Kontrola funkcionalnosti i pristup podacima putem REST API-ja.
- **WSGI Gunicorn:** Pokretanje Flask aplikacije u produkcijskom okruženju.
- **Nginx:** Web server i reverse proxy za upravljanje HTTP zahtjevima, raspodjelu opterećenja i optimizaciju performansi streaminga podataka.




### *Literatura*


1. *AdityaPratapBhuyan. (2023, October 4). Unlocking the Power of Functional Programming. DEV Community. https://dev.to/adityapratapbh1/unlocking-the-power-of-functional-programming-a-comprehensive-guide-506e*
2. *GeeksforGeeks. (2024, September 2). Functional Programming Paradigm. GeeksforGeeks. https://www.geeksforgeeks.org/functional-programming-paradigm/*
3. *Karasek, D. (2024, April 18). Unleashing the power: The advantages of functional programming. Scalac. https://scalac.io/blog/unleashing-the-power-the-advantages-of-functional-programming-in-the-digital-age/*
4. *Team, C. O. (2023, November 28). Declarative vs. Imperative Programming: 4 Key Differences. Codefresh. https://codefresh.io/learn/infrastructure-as-code/declarative-vs-imperative-programming-4-key-differences/*
5. *The Haskell Cabal | Overview. (n.d.). https://www.haskell.org/cabal/*

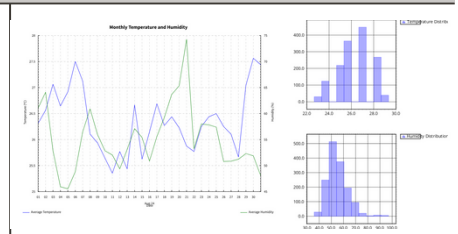

github repozitorij real-time-data-app

github repozitorij project_iot

## 02. Analiza kako funkcijsko programiranje poboljšava obradu podataka u stvarnom vremenu

- **Čiste funkcije:** Funkcije koje uvijek vraćaju isti rezultat za iste ulazne vrijednosti i nemaju nuspojave.
- **Imutabilnost podataka**: Podaci se ne mijenjaju nakon što su kreirani. Svaka promjena stvara novu instancu podataka, umjesto da se postojeći podaci mijenjaju.
- **Bezstanje operacija**: Svaka funkcija koristi samo svoje ulazne podatke i vraća rezultat bez ikakvih nuspojava. Ne utječu na globalno stanje.
- **Funckije višeg reda:** Funkcije koje mogu primati druge funkcije kao argumente ili vraćati funkcije kao rezultate. Omogućuju fleksibilnu obradu podataka.
- **Kompozicija funkcije:** Kombinacija funkcija za složene operacije.

| Funkcije | Opis | Parametri |
|---|---|---|
| fetchDataWithParams | Dohvaća podatke s API-ja prema dinamički zadanom vremenskom okviru. | apiUrl, fromDate, toDate, fromTime, toTime |
| getDataForYesterday, getDataForLast7Days, getDataForPreviousMonth | Dohvaća podatke za specifične vremenske intervale: jučerašnji dan, posljednjih 7 dana, prethodni mjesec | apiUrl |
| calculateStatistics | Izračunava osnovne statističke mjere poput prosjeka, maksimalne i minimalne vrijednosti. | data |
| average, mean, standardDeviation, coefficientOfVariation, correlation | Pomoćne funkcije za izračun statističkih mjera. | data |
| chartStatistics, chartHourlyStatistics | Prikazuje prosječne vrijednosti temperature i vlage kroz tjedan ili mjesec. | data |
| plotTemperatureHistogram, plotHumidityHistogram | Generiraju histogram raspodjele temperature i vlage. | data |
| heatIndex | Izračunava osjećaj temperature uzimajući u obzir vlažnost. | temperature, humidity |




## 03. Prednosti

- Smanjenje rizika od pogrešaka uzrokovanih promjenom stanja.
- Stvaranje malih, čistih funkcija za lakše održavanje.
- Paralelno i istodobno izvršavanje zadataka.
- Bolje testiranje i efikasnost zbog jakog sustava tipova.

## 04. Izazovi

- Učenje novog načina razmišljanja za programere.
- Manje alata i paketa za funkcijske jezike.
- Veća potrošnja memorije zbog imutabilnosti.

## 05. Pregled i usporedba

**Imperativno programiranje:** Fokusira se na sekvencijalnu obradu i manipulaciju stanja. Pristup može biti nefleksibilan i dovesti do problema s latencijom u aplikacijama zbog ovisnosti o redoslijedu izvršavanja operacija.
**Deklarativno programiranje:** Usmjereno na definiranje što treba postići, bez navođenja kako. Omogućuje bolju skalabilnost i fleksibilnost u real-time aplikacijama, osobito za filtriranje i agregaciju podataka iz više izvora.
**Objektno orijentirano programiranje (OOP):** Iako pruža strukturu kroz enkapsulaciju i nasljeđivanje, može biti teško skalabilno u aplikacijama koje obrađuju velike količine podataka zbog povećane potrošnje memorije i latencije.
**Funkcijsko programiranje:** Pruža brojne prednosti za aplikacije u stvarnom vremenu zbog nepromjenjivih podataka, paralelizacije i lijenog izvođenja. Smanjuje latenciju, poboljšava performanse i omogućuje bolju skalabilnost za velike količine podataka.