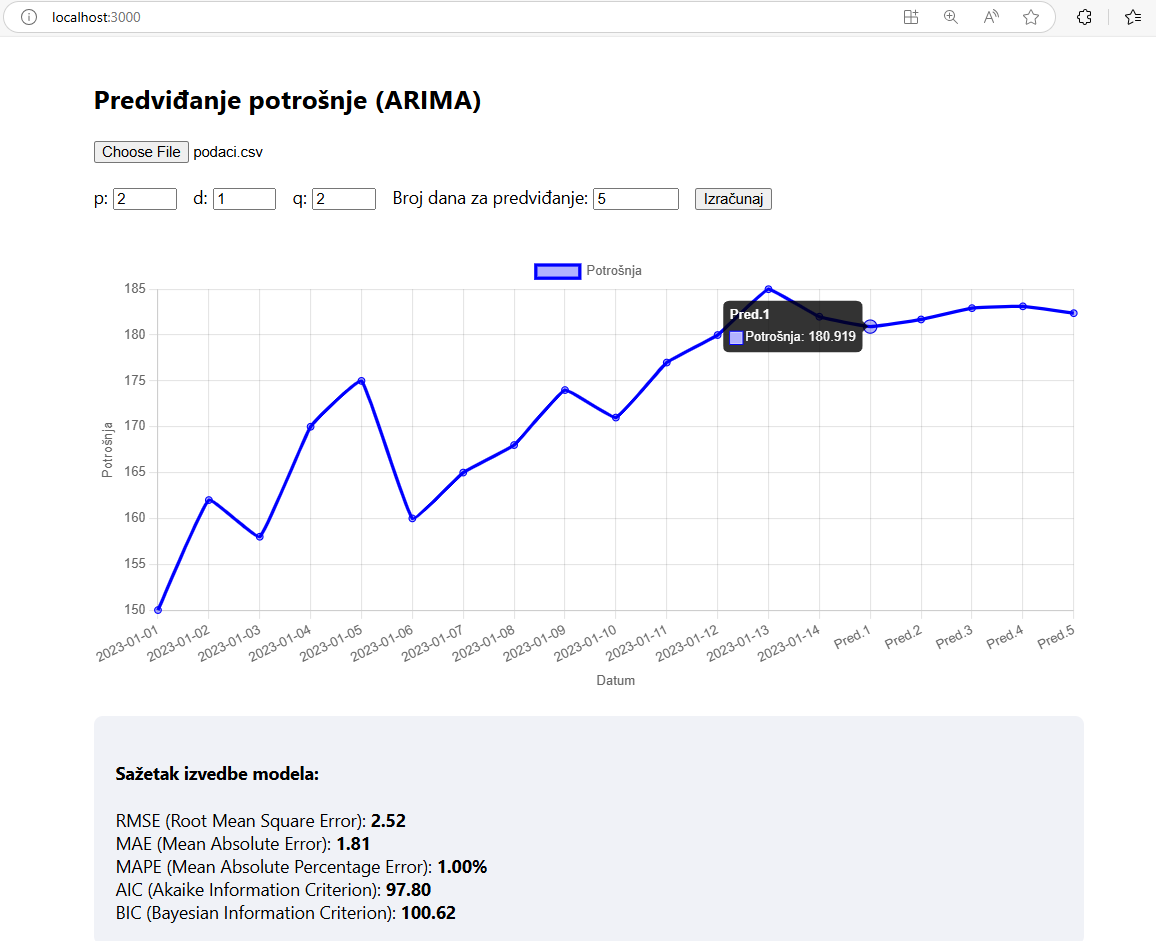
Rješenje: Aplikacija za predviđanje potrošnje

# Opis aplikacije

Predviđa potrošnju s pomoću ARIMA modela te omogućuje korisnicima vizualizaciju rezultata kroz web sučelje.

# Pregled rješenja

## 2.1. Izgled aplikacije



## 

## 2.2. Komponente rješenja

Rješenje je izgrađeno kao backend API servis i frontend aplikacija.

* Backend komponenta - Python API servis
  + Zaprima podatke za izračun predviđanja
  + Računa predviđanje s pomoću ARIMA modela
  + Računa pokazatelje sažetka izvedbe modela
  + Vraća rezultate klijentu
* Frontend komponenta - React aplikacije
  + Učitava podatke za izračun predviđanja u obliku CSV datoteke
  + Omogućuje izbor parametara ARIMA modela
  + Komunicira s Python API servisom
  + Iscrtava graf s rezultatima ARIMA predviđanja i povijesnim podacima
  + Prikazuje sažetak izvedbe modela

## 2.3. Sažetak izvedbe modela

Za sažetak se koriste pokazatelji RMSE, MAE, MAPE, AIC i BIC. U svim pokazateljima manja vrijednost je bolja, tj. RMSE, MAE i MAPE pokazuju bolju točnost, a AIC i BIC pokazuju minimalnu pogrešku uz manju složenost.

* RMSE (Root Mean Square Error) — Korijen srednje kvadratne pogreške
  + Mjeri prosječnu veličinu pogreške između predviđenih i stvarnih vrijednosti.
* MAE (Mean Absolute Error) — Srednja apsolutna pogreška
  + Mjeri prosječnu apsolutnu razliku između stvarnih i predviđenih vrijednosti.
* MAPE (Mean Absolute Percentage Error) — Srednja apsolutna postotna pogreška
  + Mjeri prosječnu apsolutnu pogrešku kao postotak stvarne vrijednosti.
* AIC (Akaike Information Criterion) — Akaikeov kriterij informacija
  + Mjeri kakvoću statističkog modela u odnosu na druge modele, uzimajući u obzir složenost modela (broj parametara).
* BIC (Bayesian Information Criterion) — Bajesov kriterij informacija
  + Slično kao AIC, ali s jačom kaznom za složenost modela.

# 

# Pokretanje aplikacije

Aplikacije se pokreće iz VS Code okoline. U nastavku su navedene programske podrške koje je potrebno instalirati i upute kako pokrenuti frontend aplikaciju i backend servis.

## 3.1. Potrebna programska podrška

* VS code - <https://code.visualstudio.com/>
* Python - <https://www.python.org/>
* Node.js - <https://nodejs.org/en>

## 3.2. Upute za pokretanje

1. Otvoriti VS Code
2. Otvoriti projekt “arima-project”
3. Otvoriti terminal 1
4. U terminalu 1 pozicionirati se u “arima-frontend” mapu
5. Pokrenuti react aplikaciju s naredbom “npm start”
6. Frontend aplikacija se otvara na adresi: <http://localhost:3000/>
7. Otvoriti terminal 2
8. U terminalu 2 pozicionirati se u “arima-project” vršnu mapu
9. Pokrenuti python servis s naredbom “python app.py”
10. Backend servis se otvara na adresi: <http://localhost:5000/>

# Korištene biblioteke

## 4.1. Python biblioteke

* Flask
  + <https://pypi.org/project/Flask/>
  + Koristimo da bi pretvorili python aplikacije u API servis
* Flask\_cors
  + <https://pypi.org/project/flask-cors/>
  + Omogućuje nam da možemo komunicirati s frontendom na drugoj domeni
* Pandas
  + <https://pypi.org/project/pandas/>
  + Koristi se za manipulaciju tabličnim podacima excel, csv,.. za uvoz podataka
* Statsmodel
  + <https://pypi.org/project/statsmodels/>
  + Koristi se za izračun predviđanja pomoću Arima statističkog modela
* Numpy
  + <https://pypi.org/project/numpy/>
  + Koristi se za izračun prikaza sažetka modela

## 4.2. Javascript biblioteke

* React
  + <https://www.npmjs.com/package/react>
  + Biblioteka za razvoj korisničkog sučelja
* Papaparse
  + <https://www.npmjs.com/package/papaparse>
  + Koristi se učitavanje i za rad csv datotekom
* Axios
  + <https://www.npmjs.com/package/axios>
  + Koristi se za slanje podataka na backend
* Chart.js
  + <https://www.npmjs.com/package/chart.js>
  + Koristi se za iscrtavanje grafova na sučelju
* react-chartjs-2 -
  + <https://www.npmjs.com/package/react-chartjs-2>
  + Omotač [Chart.js](http://chart.js) za jednostavnije korištenje u React-u