

# Integracija podataka u alatu Apache Nifi

---

Novoselec, Igor

Master's thesis / Diplomski rad

2023

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Organization and Informatics / Sveučilište u Zagrebu, Fakultet organizacije i informatike**

Permanent link / Trajna poveznica: <https://um.nsk.hr/um:nbn:hr:211:112318>

Rights / Prava: [Attribution-NonCommercial-NoDerivs 3.0 Unported](#) / [Imenovanje-Nekomercijalno-Bez prerada 3.0](#)

Download date / Datum preuzimanja: **2025-03-20**



Repository / Repozitorij:

[Faculty of Organization and Informatics - Digital Repository](#)



**SVEUČILIŠTE U ZAGREBU  
FAKULTET ORGANIZACIJE I INFORMATIKE  
VARAŽDIN**

**Igor Novoselec**

**Integracija podataka u alatu Apache NiFi  
DIPLOMSKI RAD**

**Varaždin, 2023.**

**SVEUČILIŠTE U ZAGREBU**  
**FAKULTET ORGANIZACIJE I INFORMATIKE**  
**V A R A Ž D I N**

**Igor Novoselec**

**Matični broj: 0016138176**

**Studij: Informacijski sustavi**

**Integracija podataka u alatu Apache NiFi**

**DIPLOMSKI RAD**

**Mentor**

Prof. dr. sc. Kornelije Rabuzin

**Varaždin, studeni 2023**

## **Izjava o izvornosti**

Izjavljujem da je moj diplomski rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

*Igor Novoselec potvrdio prihvaćanjem odredbi u sustavu FOI-radovi*

---

## **Sažetak**

Ovaj diplomski rad istražuje temu "Integracija podataka u alatu Apache NiFi". Apache NiFi je open-source platforma koja omogućava efikasnu obradu i protok podataka iz različitih izvora. U radu se analiziraju teorijske osnove i metodologija Apache NiFi-a, ističući njegove ključne karakteristike poput fleksibilnosti, skalabilnosti i intuitivnog korisničkog sučelja.

Kroz detaljnu analizu, rad prikazuje način konfiguriranja i upotrebe različitih komponenti i procesora unutar Apache NiFi-a kako bi se uspostavio protok podataka. Također, istražuje se primjena alata u stvarnom svijetu kroz primjere integracije podataka iz različitih izvora i njihovo usmjeravanje prema ciljnim odredištima.

Na temelju istraživanja, zaključak naglašava da je Apache NiFi moćan alat za integraciju podataka koji omogućava organizacijama da učinkovito upravljaju tokom podataka, procesiraju podatke u stvarnom vremenu te osiguravaju sigurnost i kontrolu pristupa. Preporučuje se korištenje Apache NiFi-a kao pouzdanog rješenja za optimizaciju protoka podataka i informirano donošenje odluka na temelju ažuriranih informacija.

**Ključne riječi:** Apache, Java, NiFi, integracija podataka

# Sadržaj

|  |     |
|--|-----|
| Sadržaj.....   | iii |
| 1. Uvod.....   | 1   |
| 2. Big Data.....   | 2   |
| 2.1. 3V Karakteristike.....  | 2   |
| 2.2. Arhitektura.....  | 4   |
| 2.2.1. Lambda arhitektura.....   | 5   |
| 2.2.2. Kappa arhitektura.....  | 6   |
| 2.3. Alati i tehnike.....  | 7   |
| 2.3.1. Masovno paralelno procesiranje (MPP).....                           | 7   |
| 2.3.2. NoSQL baze podataka.....  | 7   |
| 2.3.3. Alati za distribuirano pohranjivanje i obradu.....                  | 9   |
| 2.3.4. Alati za računarstvo u oblaku.....                                  | 9   |
| 2.4. Izazovi i problemi velikih podataka.....                              | 10  |
| 3. Apache NiFi.....  | 12  |
| 3.1. Apache Software Foundation.....                                       | 12  |
| 3.2. Povijest.....   | 13  |
| 3.3. Upoznavanje.....  | 14  |
| 3.3.1. Zašto NiFi?.....  | 15  |
| 3.4. Praćenje podataka.....  | 17  |
| 3.5. Izražajni jezik.....  | 19  |
| 3.6. Primjeri izražajnog jezika.....                                       | 21  |
| 4. NiFi Arhitektura.....   | 24  |
| 4.1. Komponente.....   | 26  |
| 5. NiFi Sigurnost.....   | 28  |
| 5.1. Autentikacija korisnika.....  | 29  |
| 5.1.1. Single User.....  | 30  |
| 5.1.2. Lightweight Directory Access Protocol (LDAP).....                   | 31  |
| 5.2. Autorizacija.....   | 32  |
| 5.2.1. Definiranje pristupnih politika na razini komponenti:.....          | 33  |
| 5.2.2. Dodeljivanje prava na temelju korisnika ili korisničkih grupa:..... | 36  |
| 6. Primjer kreiranja Dataflow-a.....                                       | 37  |
| 7. Integracija podataka.....   | 41  |
| 7.1.1. Kreiranje procesne grupe.....                                       | 41  |

|   |    |
|---|----|
| 7.1.2. Kreiranje procesa “GetFile” .....                  | 42 |
| 7.1.3. Kreiranje procesora “SplitText” .....              | 44 |
| 7.1.4. Kreiranje kontrolnih servisa .....                 | 45 |
| 7.1.5. Kreiranje procesora “Update Record” .....          | 47 |
| 7.1.6. Kreiranje procesora ConvertAvroToJSON .....        | 49 |
| 7.1.7. Kreiranje procesora “ReplaceText” .....            | 50 |
| 7.1.8. Kreiranje procesora “ConvertJSONToSQL” .....       | 51 |
| 7.1.9. Kreiranje procesora PutSQL.....                    | 52 |
| 8. Apache NiFi i Apache Airflow .....                     | 55 |
| 8.1. Osnovne karakteristike .....                         | 55 |
| 8.2. Različitosti.....                                    | 56 |
| 9. Prednosti nad tradicionalnim alatima i nedostaci ..... | 58 |
| 10. Zaključak .....                                       | 60 |
| Popis literature.....                                     | 61 |
| Popis slika .....   | 64 |
| Popis tablica .....                                       | 66 |
| Prilog .....  | 67 |
| 11. Instalacija .....                                     | 67 |
| 11.1. Java Development Kit(JDK).....                      | 67 |
| 11.1.1. Preuzimanje JDK-a.....                            | 67 |
| 11.1.2. Postavljanje varijabli sustava.....               | 68 |
| 11.1.3. Provjera instalacije JDK-a .....                  | 70 |
| 11.2. Apache Nifi.....                                    | 72 |
| 11.2.1. Preuzimanje Apache Nifi-a .....                   | 72 |
| 11.2.2. Instalacija.....                                  | 72 |
| 11.2.3. Provjera porta .....                              | 73 |
| 11.2.4. Pokretanje Apache Nifi-a .....                    | 75 |

# 1. Uvod

U današnjem digitalnom dobu, organizacije se suočavaju sa sve većim izazovima u upravljanju golemim količinama podataka koje generiraju njihovi unutarnji i vanjski sustavi. Ti su podaci često u različitim formatima, distribuirani na različitim mjestima i pohranjeni u različitim izvorima.

Upravljanje ovom raznolikošću podataka zahtijeva učinkovite alate i tehnike za njihovu integraciju i obradu. Postoje mnogi alati i tehnologije dostupni za integraciju podataka, ali u ovom ćemo se radu usredotočiti na jedan od najpopularnijih alata: Apache NiFi.

Apache NiFi je softver otvorenog koda koji je razvila Apache Software Foundation. Nudi moćnu i skalabilnu platformu za integraciju podataka. NiFi omogućuje prikupljanje, upravljanje, transformaciju i arhiviranje podataka iz različitih izvora putem jednostavnog i intuitivnog grafičkog korisničkog sučelja.

Cilj ovog rada je ispitati i analizirati mogućnosti i prednosti Apache NiFi alata za integraciju podataka. U ovom ćemo članku istražiti arhitekturu i ključne značajke NiFi platforme, kao i praktične slučajeve upotrebe i scenarije u kojima se NiFi može koristiti za integraciju podataka. Kombinirajući praktične primjere i teorijsku analizu, ovaj rad pruža dragocjene uvide u prednosti i izazove korištenja Apache NiFi alata za integraciju podataka.



## 2. Big Data

Big data je izraz koji se odnosi na izuzetno velike zbirke podataka koje se neprestano dodaju i brzo šire (4syth.com, 2012). Analitika velikih podataka može koristiti podatke iz različitih izvora, uključujući društvene mreže, financijska tržišta i sl. Glavne kategorije velikih podataka navedene su u nastavku:

- **Strukturirani podaci:** Strukturirani podaci su bilo koji podaci koji se mogu pristupiti, obrađivati i pohranjivati u unaprijed određenom formatu. Računalna znanost je kroz vrijeme sve učinkovitija u izumu tehnika za upravljanje i izvlačenje vrijednosti iz ovakvih podataka (pod uvjetom da je format potpuno poznat unaprijed).
- **Nestrukturirani podaci:** Izazovi koje predstavlja nestrukturirani podaci su brojni i obimni, te ih je potrebno prevladati ako se želi izvući bilo kakva vrijednost iz ogromne količine ovakvih podataka koji postoje. Primjeri heterogenih izvora podataka su izvori podataka koji sadrže obične tekstualne datoteke, kao i fotografije, videozapise i druge vrste podataka. Takvi izvori podataka obično sadrže nestrukturirane podatke.
- **Polustrukturirani podaci:** Polustrukturirani podaci sadrže i strukturirane i nestrukturirane informacije. Iako se polustrukturirani podaci mogu činiti strukturiranima, oni nisu definirani konceptom tablice koji se koristi u sustavima za upravljanje relacijskim bazama podataka (DBMS). Primjeri polustrukturiranih podataka su XML datoteke.

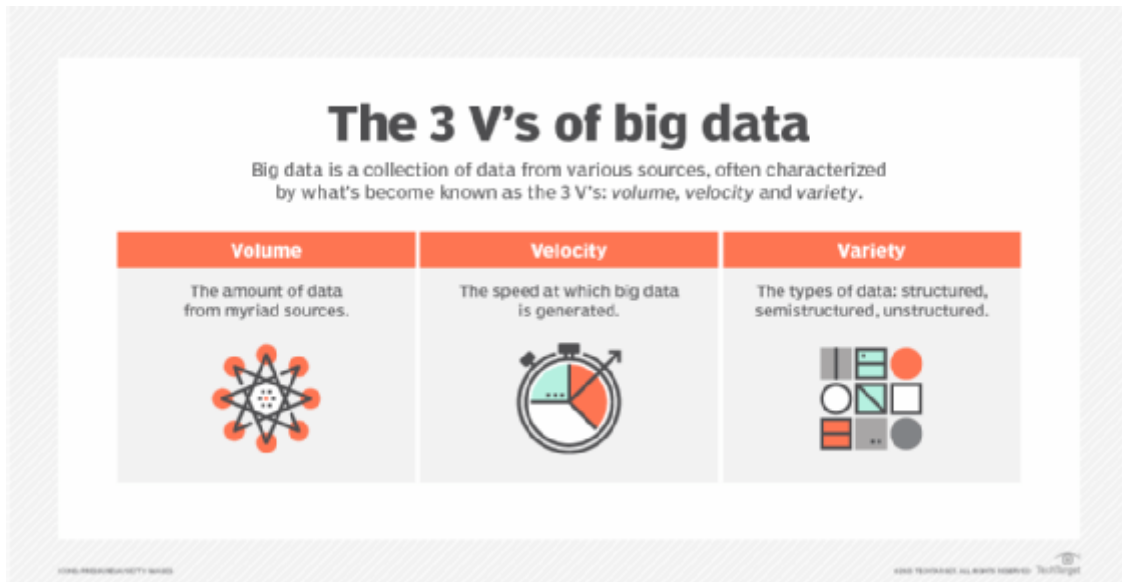
### 2.1. 3V Karakteristike

3 V-a (volumen, brzina i raznolikost) su tri ključne karakteristike velikih podataka. Volumen se odnosi na količinu podataka, brzina se odnosi na brzinu kojom se podaci obrađuju, a raznolikost se odnosi na broj vrsta podataka. 3 V-a pomažu u definiranju i mjerenju velikih podataka. Veliki podaci su podaci koji se često generiraju, u velikim količinama i u različitim oblicima. (Interviewbit.com, 2023.)

Ove karakteristike određuju tehnike modeliranja podataka koje koriste analitičari, kao što je način na koji se podaci obrađuju i pohranjuju. Oni također igraju ulogu u određivanju vrijednosti podataka. U teoriji, veća količina podataka, brzina i raznolikost su vredniji jer stvaraju jaču analitičku osnovu i pružaju dublje uvide. Podaci s malom raznolikošću mogu dovesti do pristranih analiza. Podaci niske brzine mogu biti neučinkoviti i skupi za

obradu. Podaci vrlo male količine mogu propustiti važne uzorke.

Prikupljanje i analiza podataka iz različitih izvora poboljšava donošenje odluka. Tvrtkama omogućuje bolje razumijevanje ponašanja kupaca, tržišnih trendova i uspješnosti poslovanja. Poduzeća mogu koristiti umjetnu inteligenciju i alate za strojno učenje za obradu podataka. (heavy.ai, 2023)



Slika 1: 3V karakteristike (Izvor: techtarget.com, 2023.)

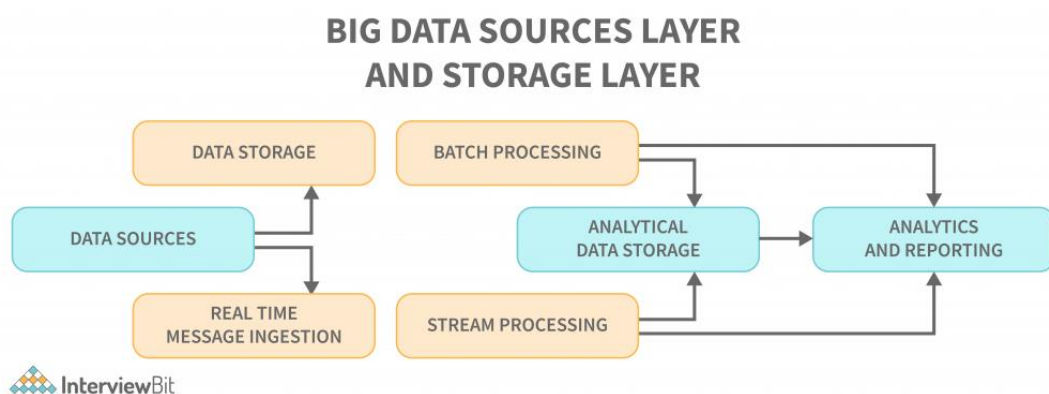
Model podataka ili arhitektura podataka koja se koristi također je važna prilikom razmatranja 3 V-a. Određene vrste modela podataka više su pogodne za obradu podataka visoke brzine. Na primjer, obrada u paketima omogućuje organizacijama brzu obradu velikih količina podataka. Ova metoda obavlja zadatke u paketima, transformira i klasificira nestrukturirane podatke kako bi se uklopili u konceptualni model podataka.

Podaci koji se nepravilno obrađuju mogu rezultirati neučinkovitostima ili netočnim predviđanjima. Na primjer, ako se numerički podaci unose u logički model podataka koji je dizajniran za kvalitativne podatke, sustav možda neće moći proizvesti rezultate ili analiza podataka može biti netočna.

## 2.2. Arhitektura

Arhitektura velikih podataka je sveobuhvatno rješenje za upravljanje ogromnom količinom podataka. Detaljno opisuje plan za pružanje rješenja i infrastrukture za rad s velikim podacima temeljenih na zahtjevima tvrtke. Jasno definira komponente, slojeve i metode komunikacije. Referentna točka je unos, obrada, pohrana, upravljanje, pristup i analiza podataka. (O'Reilly, 2015)

Arhitektura velikih podataka obično izgleda kao prikazana dolje, sa sljedećim slojevima:



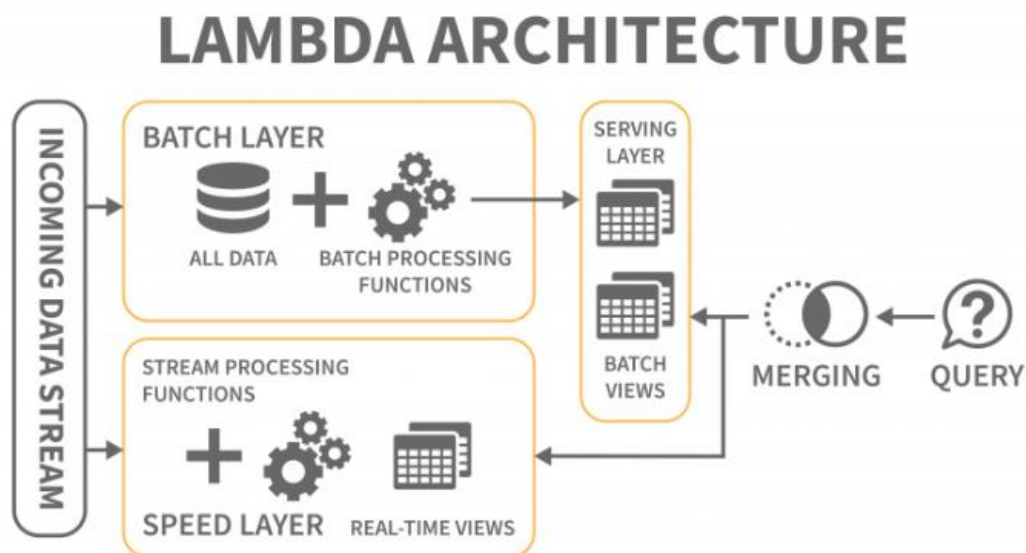
Slika 2: Big data arhitektura(Izvor: InterviewBit.com, 2023.)

- Sustav za upravljanje bazom podataka koji obrađuje unos, obradu i analizu podataka je prekompleksan ili prevelik za tradicionalnu arhitekturu. Tradicionalna arhitektura sve to obavlja odjednom, dok se sustav za upravljanje bazom podataka time bavi u dijelovima.
- Neke organizacije imaju pragove vrijednosti za gigabajte ili terabajte, dok drugima čak ni milijuni gigabajta ili terabajta nisu dovoljni.
- Na primjer, ako pogledamo sustave za pohranu, vrijednosti i troškovi pohrane su značajno smanjeni zbog ove pojave. Postoji puno podataka koji zahtijevaju različite metode pohrane.
- Arhitektura velikih podataka rješava neke od tih problema pružajući skalabilan i učinkovit način pohrane i obrade podataka. Neki od njih su podaci vezani uz batch koji se pojavljuju u određeno vrijeme i stoga se poslovi moraju rasporediti na isti način kao i batch podaci. Poslovi povezani sa streamingom zahtijevaju izgradnju stvarnog vremenskog sustava za prijenos podataka kako bi se ispunili svi njihovi zahtjevi. Taj

proces se ostvaruje putem arhitekture velikih podataka.

## 2.2.1. Lambda arhitektura

Lambda arhitektura obrađuje i batch (statične) podatke i podatke obrade u stvarnom vremenu. Koristi se za rješavanje problema izračunavanja proizvoljnih funkcija. U ovom modelu implementacije, smanjuje se latencija i zadržava se točnost uz minimalne pogreške. Arhitektura velikih podataka prikazana u nastavku slična je opisanoj:



Slika 3: Lambda arhitektura(Izvor: streamsets.com, 2023.)

Lambda arhitektura se sastoji od sljedećih slojeva:

- **Sloj batch obrade:** Sloj batch sprema dolazne podatke u njihovoj cjelokupnosti kao "batch" prikaze. "Batch" prikazi se koriste za pripremu indeksa. Podaci su nepromjenjivi, i samo se kopije originalnih podataka stvaraju i čuvaju.
- **Sloj brze obrade:** Sloj brze obrade isporučuje podatke izravno sloju batch obrade, koji je odgovoran za izračunavanje inkrementalnih podataka. Međutim, sam sloj brze obrade također može smanjiti latenciju smanjenjem broja računanja
- **Sloj posluživanja:** "Batch" prikazi i rezultati brze obrade prolaze do sloja posluživanja

kao rezultat "batch" prikaza sloja batch obrade. Sloj posluživanja indeksira prikaze i paralelizira ih kako bi osigurao brzo izvršavanje upita korisnika bez kašnjenja

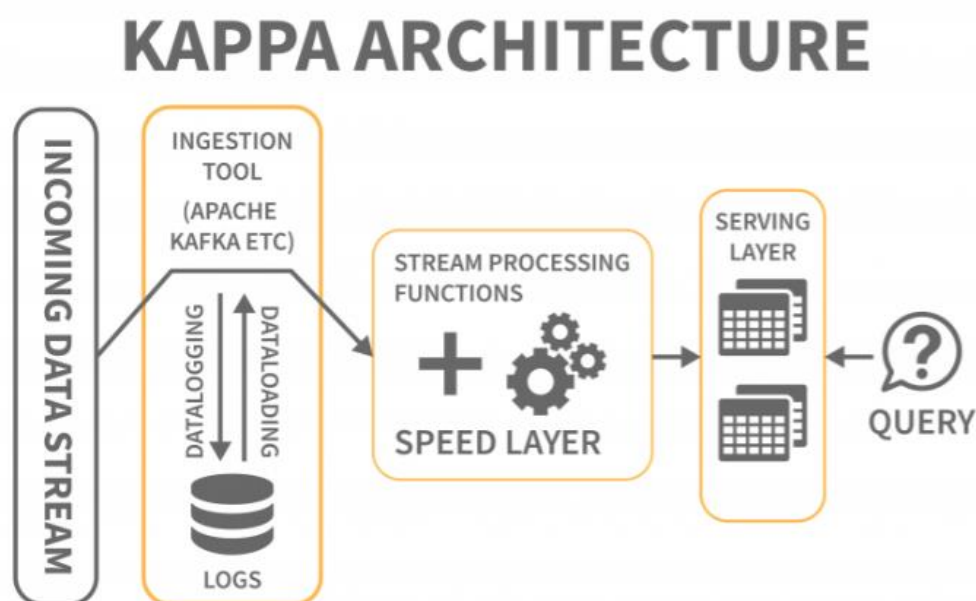
## 2.2.2. Kappa arhitektura

U usporedbi s Lambda arhitekturom, Kappa arhitektura također je namijenjena za rukovanje podacima u stvarnom vremenu i obradu podataka u "batch" načinu. Kappa arhitektura, osim što smanjuje dodatne troškove koji dolaze s Lambda arhitekturom, zamjenjuje izvor podataka s redovima poruka.

Sustavi poruka pohranjuju niz podataka u analitičkim bazama podataka, koje se zatim čitaju i pretvaraju u odgovarajući format prije nego što se spremaju za krajnje korisnike.

Arhitektura omogućuje jednostavan pristup podacima u stvarnom vremenu čitanjem i pretvaranjem podataka iz poruka u format koji je lako dostupan krajnjim korisnicima. Također pruža dodatne izlazne rezultate omogućavajući da prethodno spremljeni podaci budu uzeti u obzir.

U Kappa arhitekturi je eliminiran sloj "batch" obrade, a sloj brze obrade je poboljšan kako bi pružio mogućnost reprogramiranja. Ključna razlika u Kappa arhitekturi je da se svi podaci prikazuju kao niz ili tok. Transformacija podataka se postiže kroz "stream" mehanizam, koji je centralni mehanizam za obradu podataka.



Slika 4: Kappa arhitektura(Izvor: databricks.com, 2023.)

## 2.3. Alati i tehnike

Alat za velike podatke može se svrstati u četiri kategorije navedene u nastavku, na temelju njegove praktičnosti:

- Masovno paralelno procesiranje (MPP)
- NoSQL baze podataka
- Alati za distribuirano pohranjivanje i obradu
- Alati za računarstvo u oblaku

### 2.3.1. Masovno paralelno procesiranje (MPP)

Slabo povezani ili "shared nothing" sistem za pohranu je konstrukcija masovno paralelnog procesiranja s ciljem da se veliki broj računarskih mašina podijeli na diskretne dijelove i da se paralelno izvršavaju. MPP sistem se također naziva i slabo povezanim ili "shared nothing" sistemom. Procesiranje se vrši razdvajanjem velikog broja računarskih procesora na odvojene dijelove i njihovim paralelnim izvršavanjem.

Svaki procesor radi na odvojenim zadacima, ima drugačiji operativni sistem i ne dijeli memoriju. Također je moguće da do 200 ili više procesora radi na aplikacijama povezanim na ovu visokobrzinsku mrežu. U svakom slučaju, procesor obrađuje različite setove instrukcija i ima drugačiji operativni sistem koji se ne dijeli. MPP može također slati poruke između procesa putem sistema za razmjenu poruka koji omogućava slanje naredbi procesorima.

Baze podataka zasnovane na MPP-u su IBM Netezza, Oracle Exadata, Teradata, SAP HANA, EMC Greenplum.

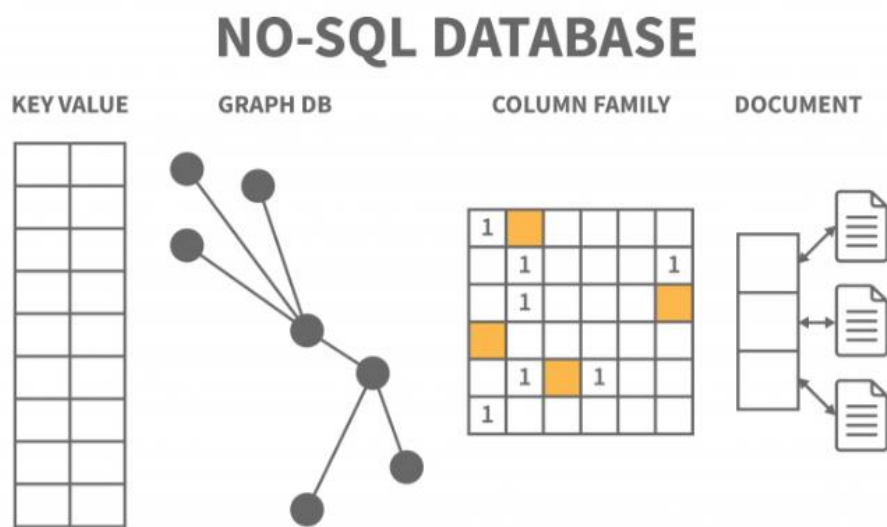
### 2.3.2. NoSQL baze podataka

Strukture se koriste za povezivanje podataka s određenom domenom. Podaci ne mogu biti pohranjeni u strukturiranoj bazi podataka sve dok se prvo ne pretvore u takvu strukturu. SQL (ili NoSQL) je ne-strukturirani jezik koji se koristi za enkapsulaciju nestrukturiranih podataka i stvaranje struktura za heterogene podatke u istoj domeni. NoSQL baze podataka nude širok spektar konfigurabilnosti, skalabilnosti i mogućnosti rukovanja velikim količinama podataka. Također postoji distribuirano skladištenje podataka koje omogućava lokalni ili

udaljeni pristup podacima.

NoSQL baze podataka obuhvaćaju sljedeće kategorije:

- Baza podataka temeljena na ključ-vrijednostima (Key-value Pair Based)
- Baza podataka temeljena na grafu (Graph based)
- Baza podataka orijentirana stupcima (Column-oriented Graph)
- Baza podataka orijentirana dokumentima (Document-oriented)



Slika 5: NO-SQL baze podataka(Izvor: InterviewBit.com, 2023.)

**Model ključ-vrijednost:** Rječnici, kolekcije i asocijativni nizovi često koriste hash tablice za pohranu podataka, ali ova baza podataka pohranjuje informacije u jedinstveni par ključ-vrijednost. Ključ je potreban za pristup podacima, a vrijednost se koristi za bilježenje informacija. Ključ je jedinstven i koristi se za dohvat i ažuriranje podataka, dok je vrijednost string, char, JSON ili BLOB. Redis, Dynamo i Riak su baze podataka ključ-vrijednost.

**Grafovski model:** Grafovske baze podataka pohranjuju entitete i veze između njih, a višestruko-relacijske. Čvorovi, veze i entiteti pohranjuju se kao elementi na grafu, a veze između tih elemenata predstavljene su bridovima (ili čvorovima). Grafovske baze podataka koriste se za mapiranje, prijevoz, društvene mreže i prostorne podatke. Mogu se koristiti i za otkrivanje uzoraka u polustrukturiranim i nestrukturiranim podacima. Grafovske baze podataka Neo4J, FlockDB i OrientDB su dostupne.

**Baza podataka orijentirana stupcima:** Za razliku od relacijskih baza podataka, imaju skup stupaca umjesto tablica. Svaki stupac je značajan i promatra se neovisno. Vrijednosti u bazi podataka pohranjene su na kontinuiran način i mogu ne imati vrijednosti. Budući da se stupcima lako pristupa, stupčane baze podataka učinkovite su u izvođenju poslova sažimanja poput ZBROJ, BROJ, AVG, MIN i MAX.

Stupčane baze podataka Cassandra, HBase i Hypertable koriste NoSQL baze podataka koje koriste stupčanu pohranu.

**Baza podataka orijentirana dokumentima:** Baza podataka orijentirana dokumentima pohranjuje dokumente kako bi bili usmjereni na dokumente umjesto na podatke. E-trgovinske aplikacije, platforme za bloganje, analitika u stvarnom vremenu, sustavi za upravljanje sadržajem (CMS) samo su neki od primjena koje koriste ove baze podataka. MongoDB, CouchDB, Amazon SimpleDB, Riak, Lotus Notes su primjeri NoSQL baza podataka orijentiranih dokumentima.

### **2.3.3. Alati za distribuirano pohranjivanje i obradu**

Raspodijeljena baza podataka je skup dijelova za pohranu podataka koji su raspodijeljeni preko mreže računala. Podatkovni centri mogu imati vlastite procesorske jedinice za raspodijeljene baze podataka. Raspodijeljene baze podataka mogu biti fizički smještene na istoj lokaciji ili raspršene po međusobno povezanoj mreži računala. Raspodijeljene baze podataka mogu biti heterogene (s različitim softverom i hardverom), homogene (s istim softverom i hardverom na svim instancama) ili različite, podržane različitim hardverom.

Vodeće platforme za obradu i distribuciju velikih podataka su Hadoop HDFS, Snowflake, Qubole, Apache Spark, Azure HDInsight, Azure Data Lake, Amazon EMR, Google BigQuery, Google Cloud Dataflow, MS SQL.

### **2.3.4. Alati za računarstvo u oblaku**

Alati za računarstvo u oblaku odnose se na usluge računarstva temeljene na mreži koje koriste razvoj i usluge interneta. Dijeljena grupa konfigurabilnih računalnih resursa koja je dostupna bilo kada i bilo gdje. Ova usluga je dostupna za plaćenu upotrebu prema potrebi i



pruža je pružatelj usluge. Platforma je vrlo korisna za obradu velikih količina podataka.

Amazon Web Services (AWS) je najpopularniji alat za računalstvo u oblaku, slijedi ga Microsoft Azure, Google Cloud, Blob Storage i DataBricks. Oracle, IBM i Alibaba također su popularni alati za računalstvo u oblaku.

## 2.4. Izazovi i problemi velikih podataka

U nastavku ćemo navesti neke izazove i probleme vezane uz koncept velikih podataka:

### 1. Nedostatak stručnjaka:

Poslovanja trebaju stručnjake za podatke kako bi učinkovito upravljali naprednom tehnologijom i ogromnim alatima za podatke na raspolaganju. Kako bi pomogli u razumijevanju ogromnih količina novih informacija koje će biti generirane kao rezultat implementacije ovih novih tehnologija, bit će zaposlen velik broj znanstvenika podataka, analitičara i inženjera. Nesposobnost pronalaska stručnih radnika s iskustvom u velikim podacima jedan je od izazova s kojima se mora suočiti svako poduzeće. Iako većina radnih profesionalaca nije svjesna ovog napretka, tehnologije koje se koriste za obradu podataka redovito se poboljšavaju.

### 2. Problemi rasta podataka:

Pronalaženje prikladnog rješenja za pohranu ogromnih količina podataka jedan je od najtežih izazova koji dolaze s radom s velikim količinama podataka. Količina podataka koja se pohranjuje u centrima za podatke i bazama podataka tvrtki raste alarmantnom brzinom. Upravljanje velikim skupovima podataka postaje sve teže kako njihove veličine nastavljaju rasti.

### 3. Odabir alata za velike podatke:

Uz toliko odgovornosti, tvrtkama je često teško odabrati najjednostavniji alat za obradu i pohranu informacija. Uspoređujući HBase i Cassandra, koji je lakši za korištenje? Koliko je Spark značajno učinkovitiji kada je riječ o pohranjivanju i analizi podataka u usporedbi s Hadoop MapReduce? Iako tvrtke traže odgovore na ova pitanja, njihovo

pronalaženje može biti teško.

#### **4. Integracija podataka iz različitih izvora:**

Tvrtka će prikupiti informacije iz različitih izvora, poput web stranica, softvera za planiranje resursa tvrtke (ERP), evidencija kupaca, financijskih izvještaja, e-pošte, prezentacija PowerPoint i sl. Moglo bi biti izazovno sastaviti izvješće sa svim tim informacijama. Zbog toga je upotreba integracije podataka prikladna unutar parametara ovog okruženja.

#### **5. Sigurnost podataka:**

Jedan od najizazovnijih problema s kojima se suočavamo u velikim podacima je osigurati sigurnost tih ogromnih skladišta podataka. Mnoge tvrtke ne prepoznaju važnost održavanja sigurnosti podataka jer su toliko zauzete obradom, pohranjivanjem i analiziranjem svojih skupova podataka. U interesu svih je izbjegavati ostavljanje skladišta podataka ranjivima ako žele spriječiti pozivanje zlonamjernih hakera. Tvrtke rizikuju gubitak do 3,7 milijuna dolara u slučaju kršenja osjetljivih informacija, poput dokumenata ili znanja.

## 3. Apache NiFi

Apache NiFi je softver otvorenog koda za obradu i distribuciju podataka koji vam omogućuje jednostavno upravljanje, transformaciju i prijenos podataka iz različitih izvora na različita odredišta.

### 3.1. Apache Software Foundation

Apache Software Foundation (ASF) američka je neprofitna organizacija koja podržava različite projekte otvorenog koda. Osnovana ga je 25. ožujka 1999. grupa programera Apache HTTP poslužitelja.



Slika 6: Logo ASF-a (Izvor: [apache.org](http://apache.org), dostupno 2023.)

Projekte Apache karakterizira proces suradničkog razvoja temeljen na konsenzusu i otvoreno i pragmatično licenciranje softvera. ASF se smatra organizacijom otvorenog koda druge generacije jer nudi komercijalnu podršku bez rizika od pada platforme. (Apache, 2023.)

ASF ima višestruku svrhu, uključujući pružanje pravne zaštite volonterima koji sudjeluju u projektima Apache kako bi se spriječilo neovlašteno korištenje zaštitnog znaka Apache od strane drugih organizacija.

ASF posjeduje različite projekte i alate, ovdje ćemo navesti nekoliko primjera:

- OpenOffice
- Apache Kafka

- Spark
- Spark SQL
- Apache NiFi

## 3.2. Povijest

Povijest Apache NiFi-a počinje razvojem alata za tokove podataka pod nazivom Niagara Files od strane NSA. NSA je izvorno razvila Niagara Files kako bi odgovorila na izazove upravljanja i obrade velikih količina podataka u složenim okruženjima. Shvativši njegov potencijal, NSA je 2014. godine odlučila otvoriti alat zajednici otvorenog koda i donirala ga Apache Software Foundationu, gdje je postao Apache NiFi.

Tijekom godina, Apache NiFi se razvijao i poboljšavao. Zajednica je pridonijela brojnim proširenjima, procesorima i integracijama, povećavajući svoju sposobnost integracije s različitim izvorima podataka, sustavima za razmjenu poruka i platformama u oblaku. Podržava integraciju s Apache Kafka, Apache Spark, Hadoop, Elasticsearch i mnogim drugim popularnim tehnologijama.

Promjene koje su napravljene u povijesti Apache NiFi-a navedene su u nastavku po godinama:

| Godina         | Opis   |
|----------------|--|
| <b>2006</b>    | Niagara Files (NiFi) razvijala je NSA 2006. više od osam godina.   |
| <b>2014</b>    | 2014. NSA ga je objavila kao softver otvorenog koda i donirala ASF-u                                       |
| <b>2015</b>    | U srpnju 2015. dostigao je status ASF projekta najviše razine i postao službeni dio Apache Project Suitea. |
| <b>Do sada</b> | Svakih 6-8 mjeseci od tada, Apache izdaje novo ažuriranje Apache NiFi.                                     |

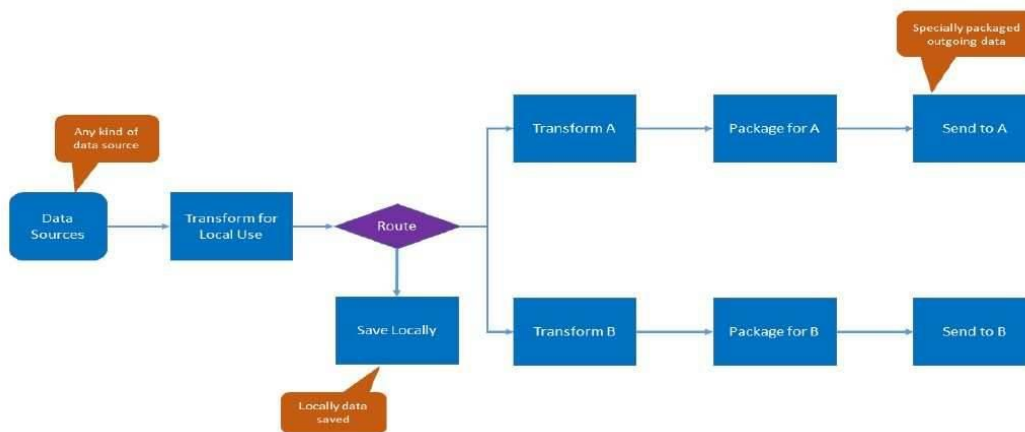
Tablica 1: Povijest Apache NiFi-a (Izvor: javatpoint.com, dostupno 2023.)

### 3.3. Upoznavanje

Kao što je ranije spomenuto, Apache NiFi je softver otvorenog koda koji je jednostavan za korištenje i pouzdan u obradi i distribuciji podataka. Omogućuje prijenos podataka iz gotovo bilo kojeg izvora na bilo koje odredište. NiFi se temelji na konfiguraciji, a ne na kodiranju, što znači da možemo kreirati ETL procese ili složene rutine obrade podataka bez potrebe za pisanjem koda. (Vishnu, 2023.) NiFi podržava agregaciju podataka i kontinuiranu obradu toka podataka. Osim toga, lako prihvaća podatke iz različitih izvora i pruža mehanizme za obradu različitih obrazaca podataka.

U osnovi, Apache NiFi je cjelovita platforma koja pokriva sljedeće aspekte:

- Funkcionalnosti koje omogućuju prikupljanje, transfer i pouzdanu isporuku podataka
- Mogućnost obrade događaja na temelju podataka s upotrebom međuspremnik i prioritarnog čekanja
- Dizajniran je da se prilagodi složenim i raznolikim tokovima podataka
- Pruža korisnički prilagođeno vizualno sučelje za razvoj, konfiguraciju i kontrolu



Slika 7: Tijek podataka (learningjournal.guru, dostupno 2023.)

### 3.3.1. Zašto NiFi?

Uz Apache NiFi kao cjelovitu platformu za upravljanje protokom podataka u poduzeću, možete koristiti unaprijed izrađeni alat za prikupljanje podataka iz različitih izvora na siguran i kontroliran način. (Craddock, 2023.) Ovo je izuzetno korisno, posebno u poslovnim scenarijima u kojima tvrtka želi prikupiti podatke iz različitih izvora za generiranje KPI-ja.

Apache NiFi pruža podršku za upravljanje tokovima podataka, posredovanje sustava i transformaciju. Postoji nekoliko razloga zašto je Apache NiFi postao popularan za rješavanje problema s podacima. Ova platforma ima mnogo značajki koje je čine jedinstvenom. Stoga su neke od značajki dostupnih na Apache NiFi platformi opisane u nastavku:



Graf: Karakteristike Apache NiFi-a (Izvor: vlastita izrada)

#### 1. UI baziran na webu

NiFi pruža web sučelje (UI) koje se može koristiti preko HTTPS-a, osiguravajući sigurnu interakciju s korisnicima. Osim toga, NiFi može obrađivati podatke u stvarnom vremenu. Također nudi intuitivan dizajn, kontrolu, nadzor i povratne informacije.

## **2. Zajamčena isporuka**

Jedna od najvažnijih i najmoćnijih značajki Apache NiFi je zajamčena isporuka podataka. To se može postići učinkovitošću upotrebom stalnog dnevnika pisanja unaprijed i repozitorija sadržaja.

## **3. Podrijetlo podataka**

NiFi nudi mehanizam za praćenje podataka koji omogućuje praćenje protoka podataka od početka do kraja. Automatski bilježi, indeksira i daje podatke o poreklu za objekte koji prolaze kroz sustav. Te podatke možemo koristiti za usklađivanje, optimizaciju, rješavanje problema i razne druge scenarije.

## **4. Proširiv**

Zahvaljujući tome, imamo priliku stvoriti vlastiti procesor, koji omogućuje brz razvoj i učinkovito testiranje. NiFi podržava sigurne protokole kao što su SSH, SSL, HTTPS i enkripciju sadržaja te nudi autorizaciju za više korisnika i upravljanje internim pravilima. Također povećava broj priključaka dostupnih u NiFi.

## **5. Interaktivno sučelje**

Tokovi podataka mogu biti prilično složeni. NiFi pruža interaktivno korisničko sučelje koje korisnicima omogućuje vizualizaciju i izražavanje tih tokova podataka. To pojednostavljuje vizualni dizajn i smanjuje složenost protoka podataka. NiFi ne samo da omogućuje vizualno modeliranje tokova podataka, već ga također izvodi u stvarnom vremenu.

## **6. Sigurnost**

Apache NiFi pruža značajku autorizacije koja pruža međusistemska, međukorisnička i višekorisnička sigurnost. Kako bi osigurao sigurnost, NiFi koristi SSL, SSH i HTTPS kao sigurne protokole. Također, koristi dodatnu enkripciju za zaštitu podataka.

### 3.4. Praćenje podataka

Praćenje svih podatkovnih aktivnosti od izvora do krajnje točke veliki je izazov za bilo koju platformu za usmjeravanje i upravljanje podacima. NiFi omogućuje korisnicima da detaljno prate sve događaje s vlastitim podacima i automatski i vrlo precizno bilježe svaki korak. Svaki pojedini događaj kao što je stvaranje, umetanje, adresiranje, modificiranje, označavanje, pregledavanje, izvoz ili brisanje podataka bilježi se s vremenom događaja, identitetom komponente koja je izvršila radnju, odredištem podataka i svim učinjenim promjenama.

NiFi Data Provenance

Displaying 1,000 of 1,000  
Oldest event available: 07/14/2016 20:56:52 UTC  
Showing the most recent 1,000 of 1,000+ events, please refine the search.

Filter by component name

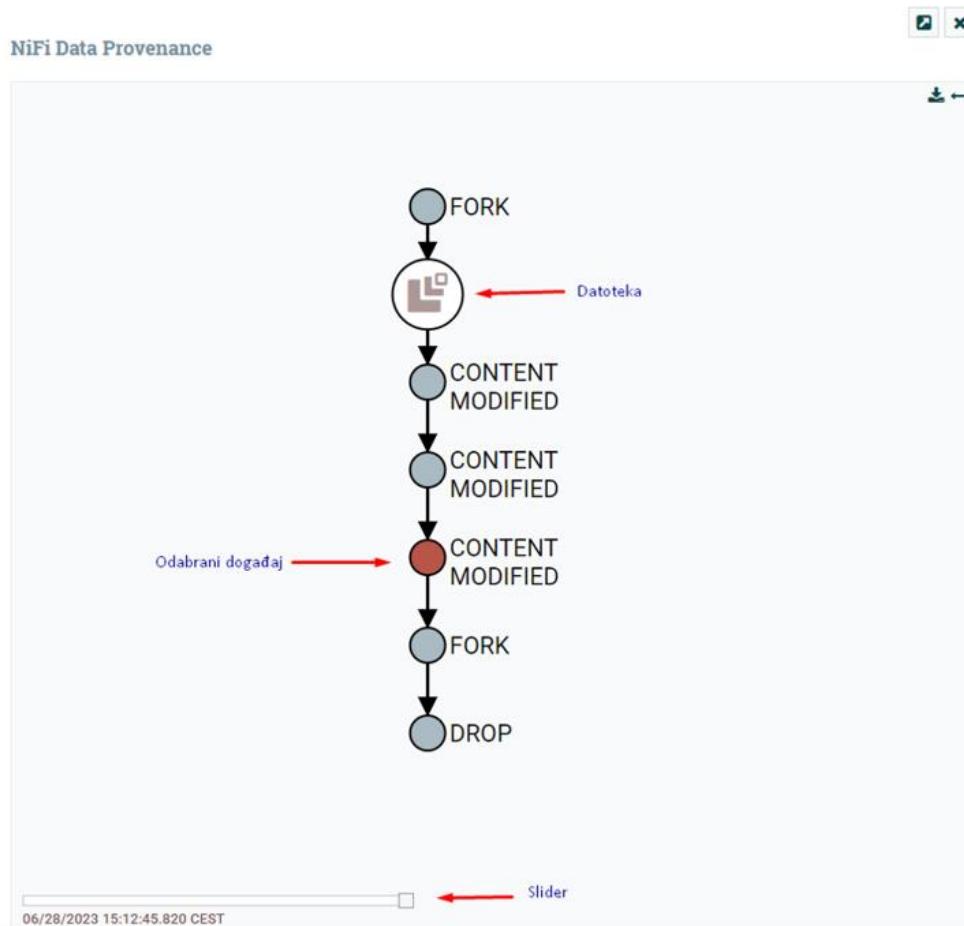
| Date/Time           | Type            | FlowFile Uuid        | Size    | Component Name    | Component Type    | Node                   |
|---------------------|-----------------|----------------------|---------|-------------------|-------------------|------------------------|
| 07/29/2016 00:14... | ATTRIBUTES_MODL | 91ae19fa-5797-45...  | 1.11 KB | Evaluate.JsonPath | Evaluate.JsonPath | nifi-05.eng.hortonw... |
| 07/29/2016 00:14... | ATTRIBUTES_MODL | bcc29546-bbd0-43...  | 1.11 KB | Evaluate.JsonPath | Evaluate.JsonPath | nifi-05.eng.hortonw... |
| 07/29/2016 00:14... | ATTRIBUTES_MODL | 9f4c3b69-6cef-40a... | 1.11 KB | Evaluate.JsonPath | Evaluate.JsonPath | nifi-05.eng.hortonw... |
| 07/29/2016 00:14... | ATTRIBUTES_MODL | 38bb2021-1f07-4e...  | 1.11 KB | Evaluate.JsonPath | Evaluate.JsonPath | nifi-05.eng.hortonw... |
| 07/29/2016 00:14... | ATTRIBUTES_MODL | f31d3aa0-40b7-46...  | 1.11 KB | Evaluate.JsonPath | Evaluate.JsonPath | nifi-05.eng.hortonw... |
| 07/29/2016 00:14... | ATTRIBUTES_MODL | 7d12c959-6952-41...  | 1.11 KB | Evaluate.JsonPath | Evaluate.JsonPath | nifi-05.eng.hortonw... |
| 07/29/2016 00:14... | ATTRIBUTES_MODL | 93f31b5c-be89-49e... | 1.11 KB | Evaluate.JsonPath | Evaluate.JsonPath | nifi-05.eng.hortonw... |
| 07/29/2016 00:14... | ATTRIBUTES_MODL | a1e5a6a4-b44e-4e...  | 1.11 KB | Evaluate.JsonPath | Evaluate.JsonPath | nifi-05.eng.hortonw... |
| 07/29/2016 00:14... | ATTRIBUTES_MODL | cf2095c8-052a-47...  | 1.11 KB | Evaluate.JsonPath | Evaluate.JsonPath | nifi-05.eng.hortonw... |
| 07/29/2016 00:14... | ATTRIBUTES_MODL | c0db8381-6c13-42...  | 1.11 KB | Evaluate.JsonPath | Evaluate.JsonPath | nifi-05.eng.hortonw... |
| 07/29/2016 00:14... | ATTRIBUTES_MODL | 9da3e06d-9715-46...  | 1.11 KB | Evaluate.JsonPath | Evaluate.JsonPath | nifi-05.eng.hortonw... |
| 07/29/2016 00:14... | ATTRIBUTES_MODL | 18247e64-41b3-4f...  | 1.11 KB | Evaluate.JsonPath | Evaluate.JsonPath | nifi-05.eng.hortonw... |

Last updated: 00:14:35 UTC

Slika 8: Praćenje podataka preko tablice (Izvor: cloudera.com, dostupno 2023.)

Gornja slika prikazuje mogućnost provjere izvora podataka za svaki procesor i vezu unutar NiFi platforme. Klikom na određenu komponentu otvorit će se tablica događaja koji odgovaraju našim kriterijima pretraživanja. Ikona izvora u prikazu tablice omogućuje nam ne samo pregled detalja početnog događaja, već i pregled izvora same datoteke toka.





Slika 9: Praćenje podataka pomoću grafa (Izvor: vlastita izrada)

Na taj način dobivamo vizualni prikaz detaljnih događaja koji su se dogodili tim podacima dok su prolazili kroz sustav. Kursorom u donjem lijevom kutu možemo vidjeti razdoblje u kojem su se ti događaji dogodili. Pomicanjem klizača lijevo-desno možemo identificirati događaje koji su uzrokovali kašnjenje sustava i tako bolje razumjeti područja kojima treba dodijeliti više resursa, kao što je broj paralelnih zadataka za CPU.

U kasnijim verzijama NiFi platforme pojavila se i mogućnost reprodukcije streaming datoteke. Sve dok izvorni podaci nisu arhivirani i pridruženi sadržaj i dalje postoji u repozitoriju sadržaja, možemo reproducirati bilo koju datoteku s bilo kojeg mjesta. Ova značajka ponavljanja uvelike pojednostavljuje razvojni ciklus protoka dopuštajući da kontinuirano poboljšavate svoj protok dok se vaši podaci kreću kroz različite grane. Krajnji korisnik tako može razumjeti točan kontekst u kojem je datoteka toka obrađena. Korisnik ima mogućnost razvijati tijekom iterativno i obrađivati podatke točno onako kako je predviđeno.

## 3.5. Izražajni jezik

NiFi izražajni jezik uvijek koristi početne oznake `${` i završne oznake `}`. Unutar ovih oznaka nalazi se sam tekst izraza. U najosnovnijem obliku, izraz može sadržavati samo naziv atributa, kao što je `${filename}`, koji vraća vrijednost atributa "filename".

U složenijem primjeru, možemo manipulirati tom vrijednošću. Na primjer, `${filename:toUpper()}` će vratiti verziju naziva datoteke ispisanu velikim slovima pomoću funkcije `toUpper`. Poziv funkcije sastoji se od pet elemenata. Prvo, imamo oznaku poziva funkcije `:`. Drugi element je naziv funkcije - u ovom slučaju, "toUpper". Slijedi otvorena zagrada `(`, a nakon toga slijede argumenti funkcije. Potrebni argumenti ovise o konkretnoj funkciji koja se koristi. U ovom primjeru koristimo funkciju `toUpper` koja ne zahtijeva argumente, pa je taj element izostavljen. Konačno, zatvorena zagrada `)` označava kraj poziva funkcije. Postoji mnogo različitih funkcija podržanih u NiFi izražajnom jeziku koje omogućuju postizanje različitih ciljeva. Nekim funkcijama se manipulira nizovima (tekstom), kao što je `toUpper` funkcija. Druge funkcije, poput funkcija jednakosti i podudaranja, koriste se za usporedbu. Tu su i funkcije za rad s datumima i vremenima te matematičke operacije.

| Reserved Characters   | Logic Operators   | Text Search  |
|---|---|--|
| <p>If these characters are present in attribute names they need to be quoted</p> <p><code>\$   { } ( ) [ ] . : ; / * ' (space) \t \r \n</code></p> <p>Ex. <code>\${'a:attribute name'}</code><br/><code>\${"a:attribute name"}</code></p>   | <p><code>isNull()</code> <code>\${filename:isNull()}</code></p> <p><code>notNull()</code> <code>\${filename:notNull()}</code></p> <p><code>isEmpty()</code> <code>\${literal('') :isEmpty()}</code></p> <p><code>equals(string)</code> <code>\${filename :equals('value')}</code></p> <p><code>equalsIgnoreCase (string)</code> <code>\${filename :equalsIgnoreCase('v')}</code></p> <p><code>gt(number)</code> <code>\${fileSize:gt(64)}</code></p> <p><code>ge(number)</code> <code>\${fileSize:ge(64)}</code></p> <p><code>lt(number)</code> <code>\${fileSize:lt(64)}</code></p> <p><code>le(number)</code> <code>\${fileSize:le(64)}</code></p> <p><code>and(bool)</code> <code>\${fileSize:gt(1) :and({fileSize:lt(4)})}</code></p> <p><code>or(bool)</code> <code>\${fileSize:lt(1) :or({fileSize:gt(4)})}</code></p> <p><code>not()</code> <code>\${filename :endsWith('sv'):not()}</code></p> <p><code>ifElse ('true val', 'falseval')</code> <code>\${filename :endsWith('csv') :ifElse('is csv', 'is not csv')}</code></p> | <p><code>filename&gt;equals('fizz buzz bazz.txt')</code></p> <p><code>startsWith (string)</code> <code>\${filename :startsWith('fizz')}</code></p> <p><code>endsWith (string)</code> <code>\${filename :endsWith('txt')}</code></p> <p><code>Contains (string)</code> <code>\${filename :contains('buzz')}</code></p> <p><code>in(string, string...)</code> <code>\${literal('NO') :in('NO', 'NOT')}</code></p> <p><code>indexOf(string)</code> <code>\${filename :indexOf('buzz')} == 5</code></p> <p><code>lastIndexOf (string)</code> <code>\${filename :lastIndexOf('z')} == 13</code></p> <p><code>find(regex)</code> <code>\${filename:find('.*zz')}</code></p> <p><code>matches(regex)</code> <code>\${filename :matches('fizz.*txt')}</code></p> <p><code>jsonPath(path)</code> <code>\${theJson :jsonPath('\$\$.attribute')}</code></p> |
| Type Conversion   |   |  |
| <p>Coerces from one format to another</p> <p><code>toString()</code> <code>\${literal(2):toString() :equals('2')}</code></p> <p><code>toNumber()</code> <code>\${literal('2'):toNumber() :equals(2)}</code></p> <p><code>toDecimal()</code> <code>\${fileSize:toDecimal()}</code></p>   | <p><b>Date/Time</b></p> <p>format is the java <a href="#">SimpleDateFormat</a></p> <p><code>format(format, zone)</code> <code>\${aDate :format('yy/MM/dd','GMT')}</code></p> <p><code>toDate (format, zone)</code> <code>\${literal('99/12/31') :toDate('yy/MM/dd','GMT')}</code></p> <p><code>now()</code> <code>\${now():toNumber() milliseconds since epoch}</code></p>  | <p><b>Utilities</b></p> <p>These subjectless functions provide useful utilities.</p> <p><code>ip()</code> local ip</p> <p><code>hostname(bool)</code> <code>\${hostname(true) fully qualified hostname}</code></p> <p><code>UUID()</code> unique generated UUID</p> <p><code>nextInt()</code> system wide counter, not maintained through restart</p> <p><code>literal(value)</code> <code>\${literal(2):gt(1)}</code></p> <p><code>getStateValue (key)</code> <code>\${getStateValue('hash')}</code></p> <p><code>thread()</code> Thread name</p>   |
| Mathematical  |   |  |
| <p><code>plus()</code> <code>\${fileSize:plus(10)}</code></p> <p><code>minus()</code> <code>\${fileSize:minus(10)}</code></p> <p><code>multiply()</code> <code>\${fileSize:multiply(10)}</code></p> <p><code>divide()</code> <code>\${fileSize:divide(10)}</code></p> <p><code>mod()</code> <code>\${fileSize:mod(10)}</code></p> <p><code>toRadix()</code> <code>\${fileSize:toRadix(10)}</code></p>                                 |   |  |
| Encode/Decode Functions   |   |  |
| <p><code>\${message:function()}</code></p> <p>Functions: <code>escapeJson</code>, <code>escapeXml</code>, <code>escapeCsv</code>, <code>escapeHtml3</code>, <code>escapeHtml4</code>, <code>unescapeJson</code>, <code>unescapeXml</code>, <code>unescapeCsv</code>, <code>unescapeHtml3</code>, <code>unescapeHtml4</code>, <code>urlEncode</code>, <code>urlDecode</code>, <code>base64Encode</code>, <code>base64Decode</code></p> |   |  |

Slika 10: NiFi izražajni jezik (Izvor: apache.org, dostupno 2023.)

Kao što je gore prikazano, kada koristimo funkciju na atributu, taj atribut nazivamo subjektom funkcije, jer je taj atribut entitet na kojem se funkcija izvodi. Dakle, možemo ulančati više funkcija zajedno, s rezultatom prve funkcije koja postaje predmetom druge funkcije i njezin rezultat postaje subjektom treće funkcije, i tako dalje. Nastavljajući naš primjer, možemo lančano povezati više funkcija u `$(filename:toUpper():equals('HELLO.TXT'))`. Broj funkcija koje se mogu kombinirati je neograničen.

Postoje i funkcije koje nemaju subjekt. Ove se funkcije pozivaju jednostavno navođenjem funkcije na početku izraza, poput `$(hostname())`. Ove se funkcije još uvijek mogu kombinirati. Na primjer `$(hostname():toUpper())`.

| String Manipulation  | Multiple Attributes  |
|--|--|
| Examples use filename equal to 'fizz buzz bazz.txt'  |  |
| <code>toUpper()</code> <code>\$(filename:toUpper())</code>   | <code>anyAttribute('attr1','attr2'...)</code> <code>\$(anyAttribute('bizz','bazz') :contains('value'))</code>                            |
| <code>toLower()</code> <code>\$(filename:toLower())</code>   | <code>allAttributes('attr1','attr2'...)</code> <code>\$(allAttributes('bizz','bazz') :contains('value'))</code>                          |
| <code>trim</code> <code>\$(literal('abc '):trim())</code> 'abc'  | <code>anyMatchingAttribute(regex)</code> <code>\$(anyMatchingAttributes('b.*zz') :contains('value'))</code>                              |
| <code>substring(start,end)</code> <code>\$(filename:substring(0, 3))</code> 'abc'  | <code>allMatchingAttributes(regex)</code> <code>\$(allMatchingAttributes('b.*zz') :contains('value'))</code>                             |
| <code>substringBefore(string)</code> <code>\$(filename:substringBefore('zz'))</code> 'fi'  | <code>anyDelineatedValue(value, delimiter)</code> <code>\$(anyDelineatedValue( \${literal('a-b-c')}, '-' ):contains('a'))</code>         |
| <code>substringBeforeLast (string)</code> <code>\$(filename:substringBeforeLast('zz'))</code> 'fizz buzz ba'                                   | <code>allDelineatedValues(value, delimiter)</code> <code>\$(allDelineatedValues( \${literal('a-b-c')}, '-' ):contains('a')) false</code> |
| <code>substringAfter(string)</code> <code>\$(filename:substringAfter('zz'))</code> ' buzz bazz.txt'  | <code>join(string)</code> <code>\$(allAttributes('attr1','attr2') :join(', '))</code>  |
| <code>substringAfterLast (string)</code> <code>\$(filename:substringAfterLast('zz'))</code> '.txt'   | <code>count()</code> <code>\$(allMatchingAttributes('b.*zz') :count())</code> number of matching   |
| <code>getDelimitedField(index, delimiter, quote char, escape char, strip char)</code> <code>\$(filename:getDelimitedField(2, ' '))</code> buzz |  |
| <code>append(string)</code> <code>\$(filename:append('.bck'))</code> 'fizz buzz bazz.txt.bck'  |  |
| <code>prepend(string)</code> <code>\$(filename:prepend('a '))</code> 'a fizz buzz bazz.txt'  |  |
| <code>replace(search,replace)</code> <code>\$( filename:replace(' ','_'))</code> 'fizz_buzz_bazz.txt'  |  |
| <code>replaceFirst(search, replace)</code> <code>\$(filename:replace(' ','_'))</code> 'fizz_buzz_bazz.txt'                                     |  |
| <code>replaceAll(regex, replace)</code> <code>\$(filename:replaceAll ('(\\w{4})\\s', '!\$1'))</code> '!fizz!buzzbazz.txt'                      |  |
| <code>replaceNull(replace)</code> <code>\$(idonotexist:'abc') :replaceNull('abc'))</code> 'abc'  |  |
| <code>replaceEmpty(replace)</code> <code>\$(literal('')):replaceEmpty('abc'))</code> 'abc'   |  |
| <code>length</code> <code>\$(filename:length())</code> 18  |  |

Slika 11: NiFi izražajni jezik (Izvor: apache.org, dostupno 2023.)

Često ćemo morati uspoređivati vrijednosti dvaju različitih atributa. To možemo postići korištenjem ugrađenih izraza. Na primjer, možemo provjeriti jednakost atributa "filename" i atributa "uuid" pomoću izraza `$(filename : equals($(uuid))`. Primijetite da se između otvarajuće zagrade za metodu equals i ugrađenog Expression nalazi razmak. Taj razmak nije obavezan i ne utječe na procjenu izraza. Svrha mu je poboljšati čitljivost izraza. Razmak između graničnika nije bitan u Expression Language-u. Dakle, možemo koristiti izraze `$( filename : equals($( uuid))` ili `$(filename:equals($(uuid))` i oba izraza znače isto. Međutim, ne možemo koristiti `$(file name:equals($(uuid))`, jer to bi tumačilo "file" i "name" kao odvojene tokene umjesto jedinstvenog tokena "filename".

## 3.6. Primjeri izražajnog jezika

U nastavku će biti prikazano par primjera korištenja izražajnog jezika unutar Apache NiFi-a:

### 1. Formatiranje datuma

Apache NiFi nudi niz značajki oblikovanja datuma. Pomoću procesora "UpdateRecord", koji je korišten i za promjenu formata datuma u praktičnom dijelu rada, lako unosimo naredbe.

Nekoliko primjera kako oblikovati datume koristeći NiFi Expression Language:

#### **Pretvaranje UNIX vremenske oznake u određeni format datuma:**

- Unos: 1618477200000 (UNIX vremenska oznaka)
- Izraz: `${field.value:toDate():format('yyyy-MM-dd HH:mm:ss')}`
- Izlaz: 2021-04-15 00:00:00

#### **Pretvaranje niza datuma u jednom formatu u drugi format:**

- Unos: 2021-04-15 00:00:00 (format datuma unosa: gggg-MM-dd HH:mm:ss)
- Izraz: `${field.value:toDate('yyyy-MM-dd HH:mm:ss'):format('dd/MM/yyyy')}`
- Izlazak: 15.04.2021

#### **Izdvajanje specifičnih komponenti datuma:**

- Unos: 2021-04-15 00:00:00
- Izraz: `${field.value:toDate('yyyy-MM-dd HH:mm:ss'):toNumber('MM')}`
- Izlaz: 4

Ovaj primjer koristi funkciju "toNumber()" za izdvajanje mjeseca iz datuma.

## 2. Promjena sadržaja unutar dokumenta

Koristeći "ReplaceText" procesor i NiFi izražajni jezik, možemo dio sadržaja dokumenta zamijeniti vlastitim. Evo nekoliko primjera korištenja jezika izraza u procesoru "ReplaceText":

- **Jednostavna zamjena teksta:**

- Vrijednost pretraživanja: "hello"
- Zamjenska vrijednost: "world"
- Strategija zamjene: doslovno
- Izraz: `${field.value:replaceAll('hello', 'world')}`

- **Korištenje atributa datoteke toka u zamjenskoj vrijednosti:**

- Vrijednost pretraživanja: "\${filename}"
- Zamjenska vrijednost: "Processed\_\${filename}"
- Strategija zamjene: doslovno
- Izraz: `${field.value:replaceAll("\\${filename}", "Processed_${filename}")}`

## 3. IF Uvjet

U Apache NiFi-u, IF uvjet se može koristiti u različitim procesorima kao što su ReplaceText, UpdateRecord, ExecuteScript itd. U nastavku ćemo prikazati i objasniti nekoliko IF uvjeta.

- **Uvjetno vrednovanje s više uvjeta:**

- Izraz: `${field.value>equals('John') and field2.value:startsWith('A')}`

Objašnjenje: Ovaj izraz provjerava je li vrijednost "field.value" jednaka 'John' I je li vrijednost "field2.value" započinje s 'A'.

- **Uspoređivanje numeričkih vrijednosti:**

- Izraz: `#{field.value:number} > 10`

Objašnjenje: Ovaj izraz pretvara vrijednost "field.value" u numeričku vrijednost i provjerava je li veća od 10.

- **Složeno manipuliranje nizovima znakova:**

- Izraz: `#{field.value:length} > 5 or field.value:startsWith('Hello')`

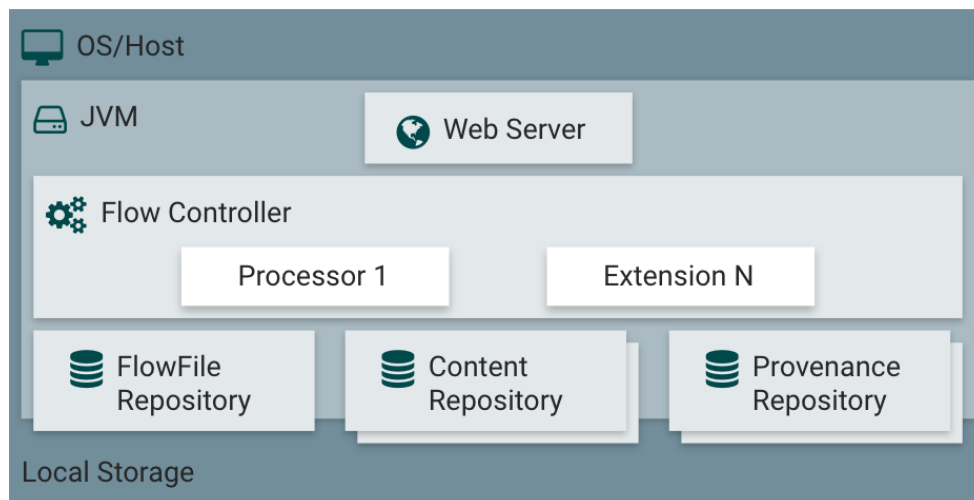
Objašnjenje: Ovaj izraz provjerava je li duljina "field.value" veća od 5 ILI započinje s 'Hello'.

- **Korištenje funkcija unutar IF izjave:**

- Izraz: `#{field.value:toLowerCase}:equals('john') or field2.value:toUpperCase}:equals('SMITH')}`

Objašnjenje: Ovaj izraz pretvara vrijednost "field.value" u mala slova i provjerava je li jednaka 'john' ILI pretvara vrijednost "field2.value" u velika slova i provjerava je li jednaka 'SMITH'. Ako je bilo koji od uvjeta istinit, izraz će biti istinit.

## 4. NiFi Arhitektura



Slika 12: Apache NiFi arhitektura (digitalis.io, dostupno 2023.)

NiFi se izvršava unutar JVM-a na glavnom operativnom sustavu. Primarne komponente NiFi-a na JVM-u su sljedeće:

### 1. Web poslužitelj

Svrha web poslužitelja je osigurati platformu za upravljanje i kontrolu NiFi API-ja temeljenog na HTTP-u.

### 2. Regulator toka

Regulator toka je glavna komponenta koja upravlja cjelokupnim procesom podataka. Pruža mehanizme za obradu podataka u obliku tokova (flows) koji se sastoje od procesora (processors), odnosno komponenti koje obavljaju različite transformacije, filtriranja i manipulacije podacima.

### 3. Ekstenzije

Postoje različite vrste NiFi ekstenzija koje su opisane u drugim dokumentima. Ključna točka ovdje je da proširenja rade i izvršavaju se unutar JVM-a.

#### 4. Repozitorij datoteke protoka

Repozitorij datoteke protoka je mjesto gdje NiFi bilježi informacije o stanju određene datoteke unutar tijeka. Repozitorij se može implementirati na različite načine. Uobičajeni pristup je uporaba stalnog Write-Ahead Loga smještenog na određenoj particiji diska.

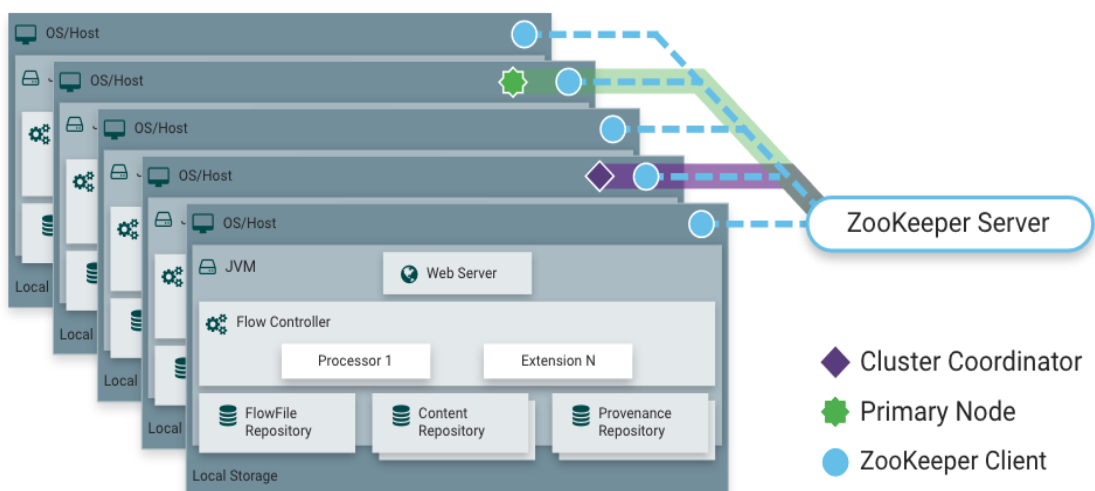
#### 5. Repozitorij sadržaja

Mjesto pohrane sadržaja je lokacija gdje se nalaze stvarni podaci sadržaja datoteke. Repozitorij se može implementirati na različite načine. Uobičajeni pristup je jednostavan mehanizam koji pohranjuje podatke u blokovima unutar datotečnog sustava. Moguće je specificirati više lokacija za pohranu kako bi se uključile različite fizičke particije i smanjio potencijalni sukob na pojedinačnom volumenu.

#### 6. Repozitorij porijekla

Mjesto pohrane podrijetla je mjesto gdje se čuvaju svi podaci o događajima podrijetla. Repozitorij se može prilagoditi korištenjem zadane implementacije koja koristi jedan ili više volumena fizičkog diska. Svaka lokacija unutar repozitorija indeksira i omogućuje pretraživanje podataka o događajima.

NiFi također može raditi unutar klastera.



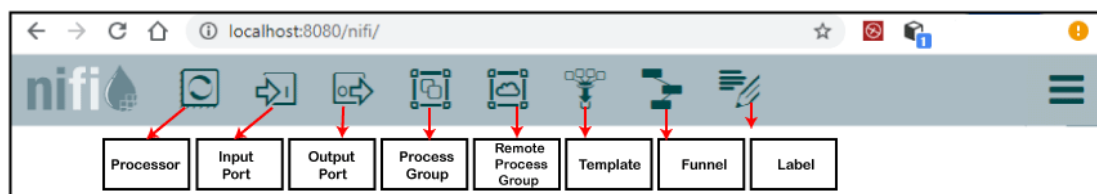


Slika 13: NiFi arhitektura (digitalis.io, dostupno 2023.)

Počevši od izdanja NiFi 1.0, koristi se paradigma klasteriranja (Clustering paradigm). Svaki čvor u NiFi klasteru obavlja iste zadatke na podacima, ali svaki radi na drugom skupu podataka. Apache ZooKeeper bira jedan čvor kao Koordinator klastera, a preuzimanje posla se automatski obavlja pomoću ZooKeeper-a. Svi čvorovi u klasteru prijavljuju informacije o statusu Koordinator klastera. Koordinator klastera je odgovoran za prekid i uspostavu veze s čvorovima.

## 4.1. Komponente

Komponente Apache NiFi građevni su blokovi protoka podataka. NiFi korisničko sučelje nudi praktičnu alatnu traku s različitim komponentama protoka podataka koje se mogu koristiti za stvaranje novog protoka.



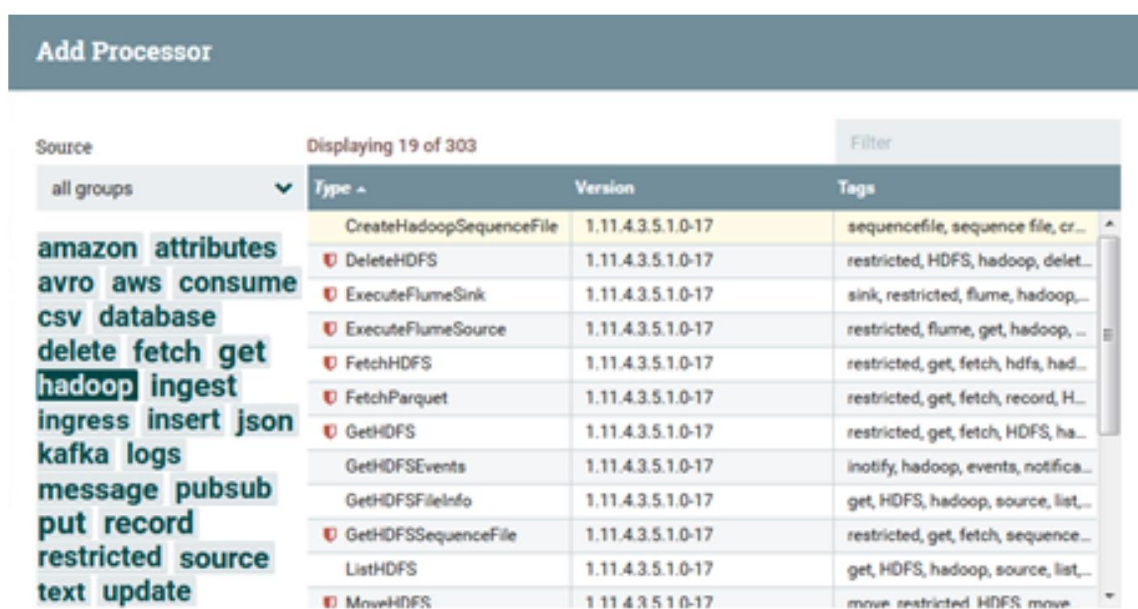
Slika 14: Apache NiFi komponente (Izvor: docs.vmware.com, dostupno 2023.)

Sljedeća tablica prikazuje svaku komponentu i njihovu glavnu svrhu.

| Komponenta            | Opis  |
|-----------------------|---|
| <b>Procesor</b>       | Procesori su osnovni blokovi za kreiranje protoka podataka  |
| <b>Ulazni port</b>    | Koristi se za dobivanje podataka od procesora koji nije prisutan u toj grupi procesa.                               |
| <b>Izlazni port</b>   | Pruža mehanizam za prijenos podataka iz grupe procesa na odredišta izvan grupe procesa                              |
| <b>Procesna grupa</b> | Koristiti se za logično grupiranje skupa komponenti tako da je protok podataka lakši za razumijevanje i održavanje. |

|                                |   |
|--------------------------------|---|
| <b>Udaljena procesna grupa</b> | Udaljena procesna grupa slična je grupi procesa, jedina razlika je što upućuje na udaljenu instancu NiFi-a. |
| <b>Lijevak(Funnel)</b>         | Lijevak je NiFi komponenta koja se koristi za kombiniranje podataka iz nekoliko veza u jednu vezu.          |
| <b>Predložak</b>               | Predložak pomaže ponovnom korištenju protoka podataka u istim ili različitim NiFi instancama.               |
| <b>Oznaka</b>                  | Oznaka se koristi za pružanje dokumentacije dijelovima tijekom podataka.                                    |

Tablica 2: Komponente Apache NiFi-a



Slika 15: NiFi procesori (Izvor: docs.vmware.com, dostupno 2023.)

NiFi uključuje mnogo različitih standardnih procesora. Svaka grupa je označena procesorima koji su povezani s tom grupom. Ovi procesori omogućuju unos podataka iz različitih sustava, usmjeravanje podataka, transformaciju, obradu, dijeljenje i agregaciju te distribuciju podataka različitim sustavima. Posebni procesori mogu se relativno lako razviti i prilagoditi ako je potrebno.

## 5. NiFi Sigurnost

Apache NiFi pruža nekoliko sigurnosnih značajki kako bi osigurao povjerljivost, integritet i dostupnost podataka koji prolaze kroz sustav. Evo ključnih aspekata sigurnosti Apache NiFi-a:

- **Autentikacija**

NiFi podržava različite mehanizme autentifikacije poput korisničkog imena/lozinke, LDAP-a, Kerberosa i klijentskih certifikata. Korisnici se moraju autentificirati prije pristupa sustavu.

- **Autorizacija**

Nakon autentifikacije, NiFi provodi politike fino granulirane kontrole pristupa temeljene na korisničkim ulogama i ovlastima. Pristupne politike određuju koje radnje korisnici mogu izvršiti na tokovima podataka, procesorima i drugim komponentama.

- **Sigurne veze**

NiFi podržava sigurnu komunikaciju putem TLS/SSL-a. Možete konfigurirati NiFi za korištenje SSL/TLS certifikata kako biste šifrirali podatke u prijenosu i provjerili identitet NiFi čvorova i klijenata.

- **Zaštita podataka**

NiFi pruža mehanizme za zaštitu osjetljivih podataka unutar sustava. Podržava šifriranje podataka u mirovanju za repozitorije i repozitorije sadržaja. Dodatno, procesori poput EncryptContent i DecryptContent mogu se koristiti za šifriranje i dešifriranje podataka dok prolaze kroz NiFi.

- **Sigurno rukovanje parametrima**

NiFi omogućuje definiranje osjetljivih svojstava kao sigurnih parametara. Ti se parametri mogu šifrirati i sigurno pohraniti, osiguravajući da osjetljive informacije poput lozinki nisu izložene u čitljivom obliku.

- **Sigurna razmjena podataka**

NiFi se integrira s vanjskim sustavima i pruža sigurne mogućnosti razmjene podataka. Podržava protokole poput HTTPS-a, SFTP-a i FTPS-a za sigurni prijenos datoteka.

- **Revizija i praćenje**

NiFi pruža detaljne mogućnosti revizije i praćenja kako bi pratili korisničke aktivnosti i događaje u sustavu. Možete konfigurirati NiFi da generira dnevničke zapise revizije koji bilježe informacije o tokovima podataka, korisničkim radnjama i događajima u sustavu.

- **Sigurna komunikacija klastera**

U klasteriziranom okruženju, NiFi osigurava sigurnu komunikaciju između čvorova klastera korištenjem sigurnih protokola i certifikata. Također pruža mehanizme za osiguravanje ZooKeeper ansambla koji se koristi za koordinaciju.

- **Siguran vanjski pristup**

NiFi se može implementirati iza vatrozida ili inverznog proxyja kako bi se kontrolirao vanjski pristup. Možete konfigurirati vatrozide ili proxyje da ograniče pristup NiFi sučeljima i portovima, osiguravajući da samo ovlašten promet dolazi do sustava.

Bitno je istaknuti da uprkos tome što Apache NiFi pruža moćne sigurnosne funkcionalnosti, presudno je pravilno podesiti i upravljati ovim karakteristikama kako bi se osigurala sveobuhvatna sigurnost implementacije NiFi.

## 5.1. Autentikacija korisnika

NiFi podržava autentikaciju korisnika korištenjem brojnih konfigurabilnih protokola i strategija. Autentifikaciju korisničkog imena i lozinke provodi 'Dobavljač identiteta za prijavu'. Davatelj identiteta za prijavu je plug-in mehanizam za autentifikaciju korisnika putem njihovog korisničkog imena/lozinke. Koji će se pružatelj identiteta za prijavu koristiti konfiguriran je u datoteci `nifi.properties`. Trenutno NiFi nudi korisničko ime/lozinku s opcijama Login Identity Providers za jednog korisnika, Lightweight Directory Access Protocol (LDAP) i Kerberos.

Svojstvo `nifi.login.identity.provider.configuration.file` navodi konfiguracijsku datoteku za pružatelje identiteta za prijavu. Prema zadanim postavkama, ovo je svojstvo postavljeno na `./conf/login-identity-providers.xml`.

Svojstvo `nifi.security.user.login.identity.provider` označava koji od konfiguriranih pružatelja identiteta za prijavu treba koristiti. Zadana vrijednost ovog svojstva je jedan korisnik-provider koji podržava provjeru autentičnosti s generiranim korisničkim imenom i lozinkom. Za

autentifikaciju s jednom prijavom, NiFi će preusmjeriti korisnike na Identity Provider prije povratka na NiFi. NiFi će zatim obraditi odgovore i pretvoriti atribute u informaciju tokena aplikacije.

Korisnik se ne može anonimno autentificirati sa zaštićenom instancom NiFi osim ako **nifi.security.allow.anonymous.authentication** nije postavljen na true. Ako je to slučaj, NiFi također mora biti konfiguriran s autorizatorom koji podržava autorizaciju anonimnog korisnika.

### 5.1.1. Single User

Zadani pružatelj identiteta za prijavu jednog korisnika podržava automatizirano generiranje vjerodajnica korisničkog imena i lozinke.

Zadana konfiguracija u nifi.properties omogućuje autentifikaciju jednog korisnika:

**nifi.security.user.login.identity.provider=single-user-provider**

Zadani **login-identity-providers.xml** uključuje definiciju pružatelja:

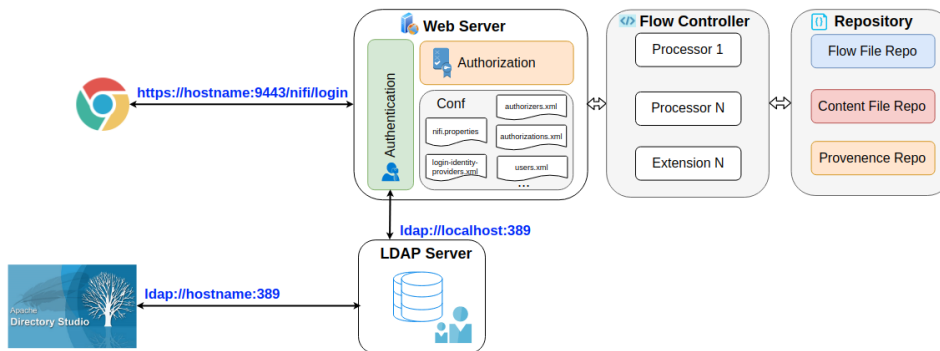
```
The 'Username' will be a random UUID and the 'Password' will be stored using bcrypt hashing
-->
<provider>
  <identifier>single-user-provider</identifier>
  <class>org.apache.nifi.authentication.single.user.SingleUserLoginIdentityProvider</class>
  <property name="Username">1cd298c6-88e2-44a2-adac-e33496eb1930</property>
  <property name="Password">$2b$12$0epjSUJWAAIbXXpJrC7EyeVrZUIv/Q0XBHSvjxUVAf/0xEjyTJHQy</property>
</provider>
<!--
```

Slika 16 : Prikaz login identifyera(Izvor: vlastita izrada)

Generirano korisničko ime bit će nasumični UUID koji se sastoji od 36 znakova. Generirana lozinka bit će nasumični niz koji se sastoji od 32 znaka i pohranjuje se korištenjem bcrypt hashiranja.

## 5.1.2. Lightweight Directory Access Protocol (LDAP)

Metoda LDAP provjere autentičnosti korisnika koristi se jer je jednostavna za upravljanje i postavljanje, a istovremeno održava siguran način.



Slika 17: LDAP arhitektura (vanducng.dev, dostupno 2023.)

Tri su glavne komponente na prethodnom dijagramu, uključujući LDAP poslužitelj, autentifikaciju i autentifikaciju:

- **LDAP poslužitelj**: upravljajte pristupom korisnika NiFi poslužitelju. Ovaj poslužitelj može se postaviti unutar istog ili zasebnog poslužitelja s NiFijem. U ovom postu koristi se biblioteka OpenLDAP kao njena popularnost.
- **Autentikacija**: omogućite značajku provjere autentičnosti za izvođenje rukovanja s LDAP poslužiteljem kako biste prepoznali tko se prijavljuje na poslužitelj.
- **Autorizacija**: razlikuje koja se pravila primjenjuju za trenutni račun za prijavu.

Postavite sljedeće u `nifi.properties` kako biste omogućili LDAP provjeru autentičnosti korisničkog imena/lozinke:

**`nifi.security.user.login.identity.provider=ldap-provider`**

Izmijenite `login-identity-providers.xml` da omogućite `ldap-provider`.

```

<provider>
<identifier>ldap-provider</identifier>
<class>org.apache.nifi.ldap.LdapProvider</class>
<property name="Authentication Strategy">START_TLS</property>

<property name="Manager DN"></property>
<property name="Manager Password"></property>

<property name="TLS - Keystore"></property>
<property name="TLS - Keystore Password"></property>
<property name="TLS - Keystore Type"></property>
<property name="TLS - Truststore"></property>
<property name="TLS - Truststore Password"></property>
<property name="TLS - Truststore Type"></property>
<property name="TLS - Client Auth"></property>
<property name="TLS - Protocol"></property>
<property name="TLS - Shutdown Gracefully"></property>

<property name="Referral Strategy">FOLLOW</property>
<property name="Connect Timeout">10 secs</property>
<property name="Read Timeout">10 secs</property>

<property name="Url"></property>
<property name="User Search Base"></property>
<property name="User Search Filter"></property>

<property name="Identity Strategy">USE_DN</property>
<property name="Authentication Expiration">12 hours</property>
</provider>

```

Slika 18: Login-identity-providers.xml(Izvor: vlastita izrada)

## 5.2. Autorizacija

Apache NiFi pruža mehanizme autorizacije kako bi kontrolirao pristup svojim resursima i operacijama. Autorizacija u NiFi se temelji na politikama koje su definirane za različite komponente i radnje unutar sustava. NiFi podržava dva tipa autorizacije:

- **Autorizacija temeljena na korisniku:**

Korisnici se mogu autentificirati koristeći različite mehanizme kao što su LDAP, Kerberos ili OpenId Connect. Nakon autentifikacije, NiFi dodjeljuje identitet korisnika sesiji. Mogu se definirati politike autorizacije kako bi se dodijelio ili odbio pristup određenim komponentama i radnjama na temelju identiteta korisnika.

- **Pristupne politike:**

Pristupne politike u NiFi se definiraju na razini komponenti. Svaka komponenta ima povezan skup radnji koje se mogu izvršiti na njoj. Pristupne politike određuju koji korisnici ili

korisničke grupe imaju dozvolu izvršavati određene radnje na određenoj komponenti. Radnje uključuju čitanje, pisanje, izmjenu, brisanje i druge.

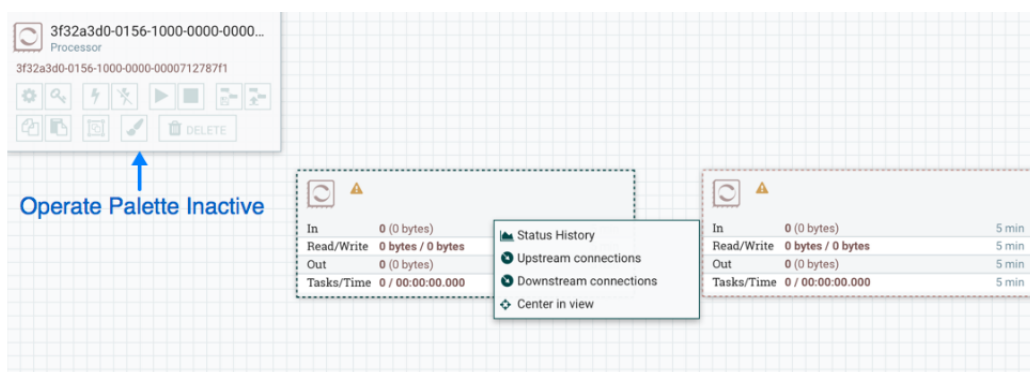
### 5.2.1. Definiranje pristupnih politika na razini komponenti:

Postavljanje pristupnih politika na razini komponente u Apache NiFi omogućuje kontrolu pristupa određenim komponentama ili radnjama u sustavu. Ovo je korisno za ograničavanje određenih korisnika ili grupa korisnika ili za dopuštanje izvođenja određenih radnji na određenim komponentama u tijeku podataka.

Na primjer, možete definirati pravila pristupa koja dopuštaju samo određenim korisnicima čitanje podataka iz određene baze podataka putem ExecuteSQL procesora. Na taj način samo korisnici s odgovarajućim dopuštenjima mogu pristupiti podacima u ovoj bazi putem NiFi-a.

Primjer definiranja politike na razini komponente prikazat ćemo na primjeru 2 korisnika. Jedan korisnik je admin, dok je drugi korisnik novokreirani račun te nema pristup nijednom dataflowu.

Postavit ćemo u Apache NiFi-u 2 procesora: GenerateFlowFile i LogAttribute. Korisnik 1 može podešavati oba procesora, dok korisnik 2 ne može pristupiti niti jednom. Postavit ćemo politiku tako da Korisnik 2 može podesiti GenerateFlowFile procesor



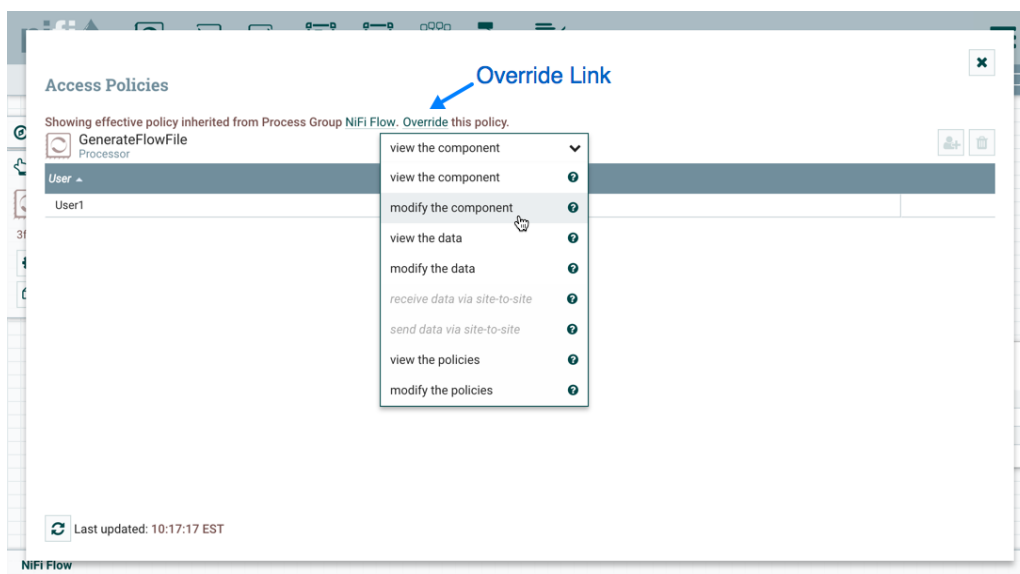
Slika 19: Prikaz procesora iz perspektive Korisnika 2 (izvor: vlastita izrada)

Da bi omogućili Korisniku 2 premještanje procesora GenerateFlowFile u dataflowu, a



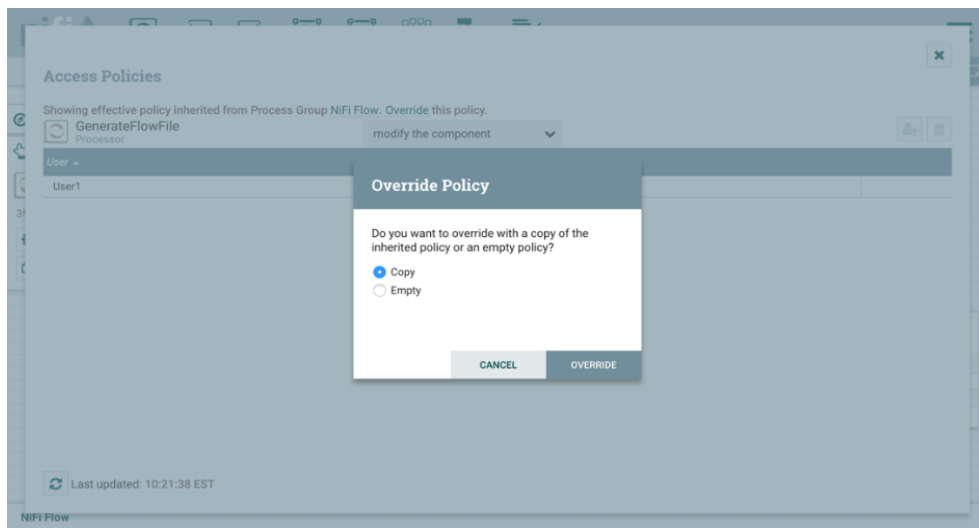
samo tog procesora, moramo izvršiti sljedeće korake:

1. Odaberite procesor GenerateFlowFile tako da bude istaknut.
2. Odaberite ikonu Pristupnih politika (Access Policies Icon) iz palete Operate i otvorit će se dijalog Pristupnih politika.
3. Izaberite "izmijeni komponentu" iz padajućeg izbornika politika. "Izmijeni komponentu" politika koja već postoji na procesoru (dijete) je "izmijeni komponentu" politika naslijeđena od korijenske procesne grupe (roditelj) na kojoj Korisnik 1 ima privilegije.



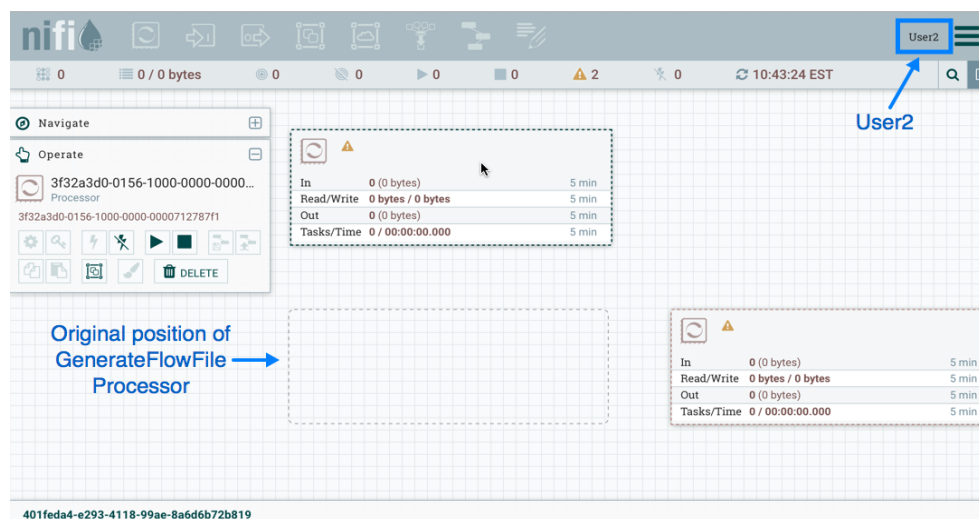
Slika 20 : Prikaz postavljanja politike(izvor: vlastita izrada)

4. Odaberite poveznicu "Override" u poruci o nasljeđivanju politike. Prilikom stvaranja zamjenske politike, dobit ćete mogućnost odabira između zamjene kopijom naslijeđene politike ili prazne politike. Odaberite gumb "Override" kako biste stvorili kopiju.



Slika 21: Override politike(izvor: vlastita izrada)

- Na zamjenskoj politici koja je stvorena, odaberite ikonu "Dodaj korisnika" (Add User Icon). Pronađite ili unesite Korisnik2 u polje "Korisnički identitet" (User Identity) i odaberite "OK". S ovim promjenama, Korisnik1 zadržava mogućnost premještanja oba procesora na platnu. Korisnik2 sada može premještatati procesor GenerateFlowFile, ali ne može premještatati procesor LogAttribute.



Slika 22: Omogućavanje politike(izvor: vlastita izrada)

Ovaj primjer pokazuje kako se pristupne politike mogu koristiti za kontrolu pristupa komponentama u Apache NiFi. Važno je napomenuti da postavke sigurnosti i definiranje pristupnih politika mogu varirati ovisno o verziji Apache NiFi-a koju koristite. Stoga je preporučljivo konzultirati dokumentaciju Apache NiFi-a za detaljnije informacije o postavljanju

pristupnih politika na razini komponenti.

### 5.2.2. Dodeljivanje prava na temelju korisnika ili korisničkih grupa:

Dodeljivanje prava na temelju korisnika ili korisničkih grupa u Apache NiFi omogućava kontrolu pristupa određenim komponentama ili akcijama na osnovu identiteta korisnika ili pripadnosti korisničkim grupama.

Primjer: Pretpostavimo da imate korisničku grupu "Analitičari" u Apache NiFi koja sadrži korisnike koji imaju pristup analitičkim procesorima. Želite omogućiti samo korisnicima iz te grupe da imaju pristup procesoru "ExecuteSQL" kako bi izvršavali upite nad bazom podataka.

Da bismo omogućili pristup samo korisnicima iz grupe "Analitičari" za procesor "ExecuteSQL" u Apache NiFi, slijedite ove korake:

- Pokrenite Apache NiFi i prijavite se kao administrator.
- Otvorite korisničko sučelje Apache NiFi.
- U postavkama sistema, konfigurirajte autentifikaciju kako biste autentificirali korisnike i dobili informacije o njihovim grupama.
- Nakon konfiguracije autentifikacije, odaberite procesor "ExecuteSQL".
- U postavkama procesora, pronađite odjeljak za sigurnost ili autorizaciju.
- U odjeljku za autorizaciju, dodajte pravilo ili politiku pristupa za procesor "ExecuteSQL".
- U pravilu ili politici pristupa, odaberite "Analitičari" kao ciljnu korisničku grupu.
- Postavite pravilo ili politiku pristupa da dozvoljava pristup samo korisnicima iz grupe "Analitičari" na procesoru "ExecuteSQL".
- Spremite promjene u postavkama procesora.
- Nakon postavljanja pravila pristupa, samo korisnici koji pripadaju grupi "Analitičari" imaju dozvolu izvršavati radnje na procesoru "ExecuteSQL".

Ovo je korisno kako bismo definirali kojim korisnicima ili grupama je dozvoljeno izvršavati određene radnje unutar Apache NiFi-a

## 6. Primjer kreiranja Dataflow-a

Nadalje imamo prikaz kreiranja osnovnog dataflow-a unutar Apache NiFi-a. Da bismo stvorili tijek podataka, povučemo komponentu procesora i isпустimo na NiFi platno i zatim povežimo s komponentom veze.

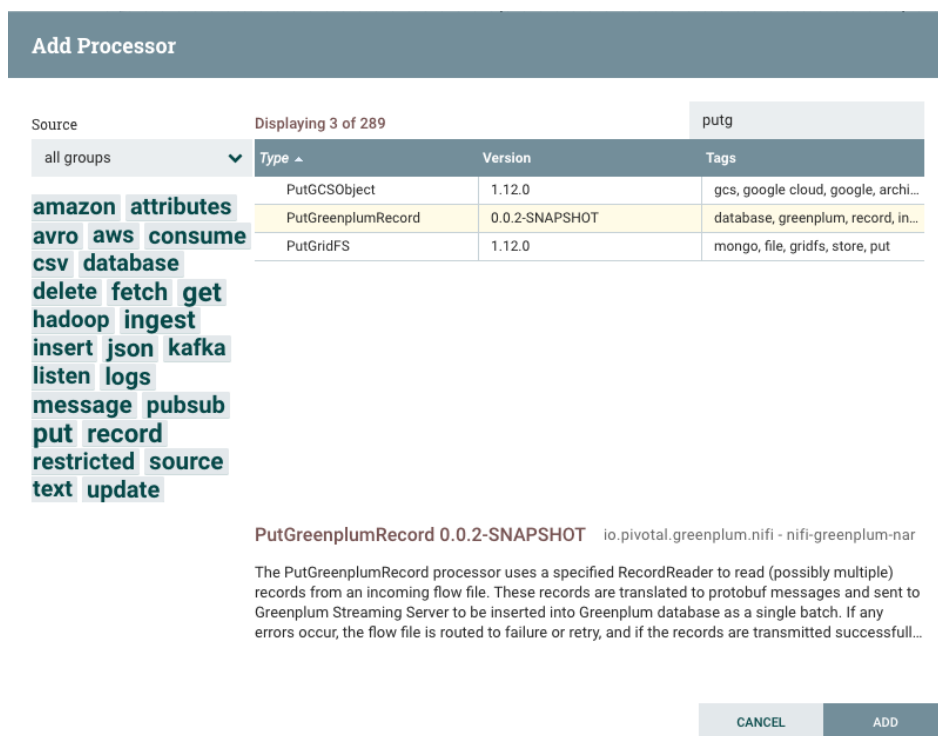


Slika 23: Ikona komponente procesora(Izvor: vlastita izrada)

### 1. Dodavanje procesora na platno:

Povucite komponentu procesora s alatne trake komponenti na platno i tamo je isпустite.

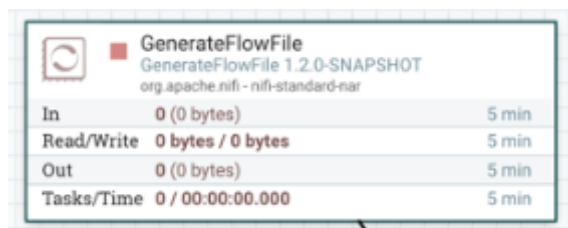
Dijaloški okvir Dodaj procesor prikazuje:



Slika 24: Dijaloški okvir procesora(Izvor: vlastita izrada)

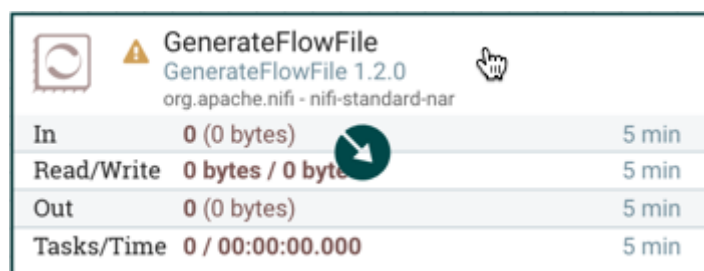
2. Odaberite procesor koji želite dodati pomicanjem kroz popis, odabirom pojma za pretraživanje iz lijevog okna ili unosom naziva procesora u polje Filtar u gornjem desnom kutu dijaloškog okvira.
3. Odaberite željeni procesor iz tablice i dvokliknite ili pritisnite DODAJ.

Dijaloški okvir Dodaj procesor se zatvara, a komponenta procesora se dodaje na platno



Slika 25: Komponenta procesora(Izvor: vlastita izrada)

Započinjemo odnos povezivanjem procesora u protok podataka. Povežimo procesore tako da zadržimo pokazivač miša iznad izvornog procesora, kliknemo ikonu veze (označena zelena strelica) u izvornom procesoru te ju povučemo i isпустimo na odredišni procesor.



Slika 26: Povezivanje komponenta(Izvor: vlastita izrada)

Prikazuje se dijaloški okvir Stvori vezu. Koristimo kartice DETALJI i POSTAVKE u ovom dijaloškom okviru za konfiguriranje veze, uključujući njezin naziv, pragove, prioritete i strategiju ravnoteže opterećenja.

### Configure Connection

DETAILS
SETTINGS

---

Name

Id  
208bf461-0166-1000-ffff-ffffeb88d70

FlowFile Expiration  
0 sec

Back Pressure  
Object Threshold 10000

Size Threshold 1 GB

Load Balance Strategy  
Do not load balance

Available Prioritizers

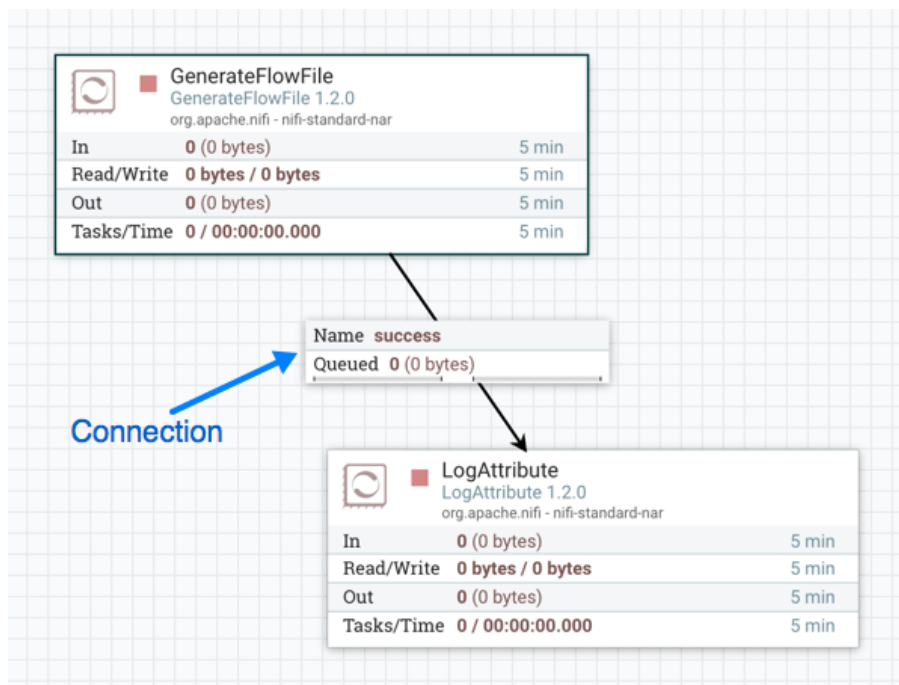
- FirstInFirstOutPrioritizer
- NewestFlowFileFirstPrioritizer
- OldestFlowFileFirstPrioritizer
- PriorityAttributePrioritizer

Selected Prioritizers

CANCEL
APPLY

Slika 27: Konfiguracija veze(Izvor: vlastita izrada)

Veza je predstavljena na NiFi platnu kao objekt između procesora, a uključuje liniju s usmjerenom strelicom od izvora do odredišta:



Slika 28: Prikaz dataflow-a(Izvor: vlastita izrada)

Komponenta procesora koju dodate na platno je u zaustavljenom stanju. Procesor mora biti omogućen i pokrenut prije nego što se može pokrenuti. Strategija raspoređivanja procesora određuje kada i kako se procesor pokreće.

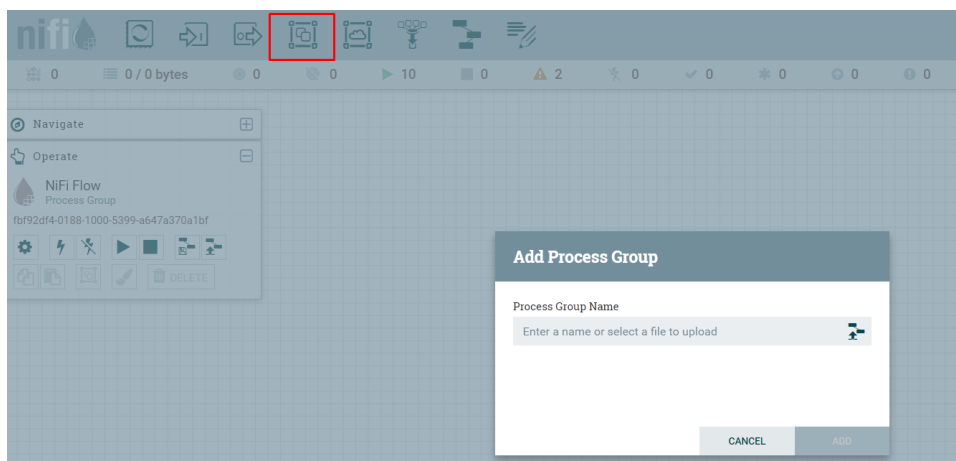
Pokrenite komponentu desnim klikom na komponentu i odabirom Start iz kontekstnog izbornika komponente. Ili pokrenite više komponenti tako da odaberete svaku komponentu koju želite pokrenuti i pritisnete gumb za pokretanje u paleti operacija.

## 7. Integracija podataka

Prikazat ćemo praktični dio rada korak po korak. Počet ćemo s učitavanjem CSV datoteke u Apache NiFi, zatim ćemo pretvoriti format iz CSV u JSON te provesti različite transformacije nad podacima. Na kraju, podatke ćemo prenijeti u MySQL bazu podataka.

### 7.1.1. Kreiranje procesne grupe

Podaci koje ćemo koristiti prikazuju potrebe diljem svijeta tijekom protekle godine. Pokrećemo Apache NiFi prema uputama u prethodnom dijelu. Za početak, ćemo dodati grupu procesa u Apache NiFi-u u kojoj ćemo izvršiti sve korake potrebne za učitavanje podataka u MySQL bazu. Pritisnemo ikonu koja je prikazana dolje i unesemo naziv grupe procesa.



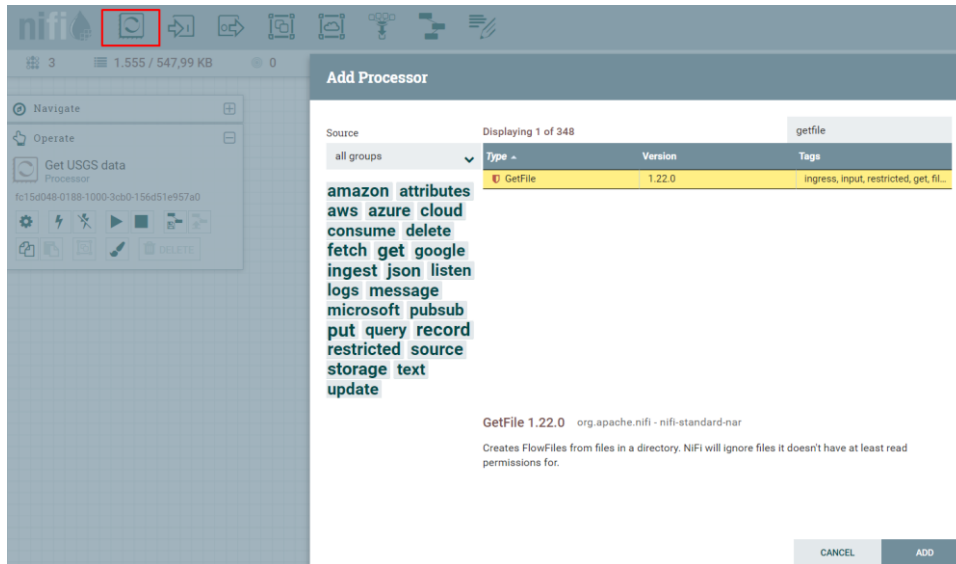
Slika 29: Dodavanje procesne grupe(Izvor: vlastita izrada)

Kada smo stvorili grupu procesa, dva puta kliknemo na nju da je otvorimo ili desnom tipkom miša kliknemo na nju i odaberemo "Enter Group".



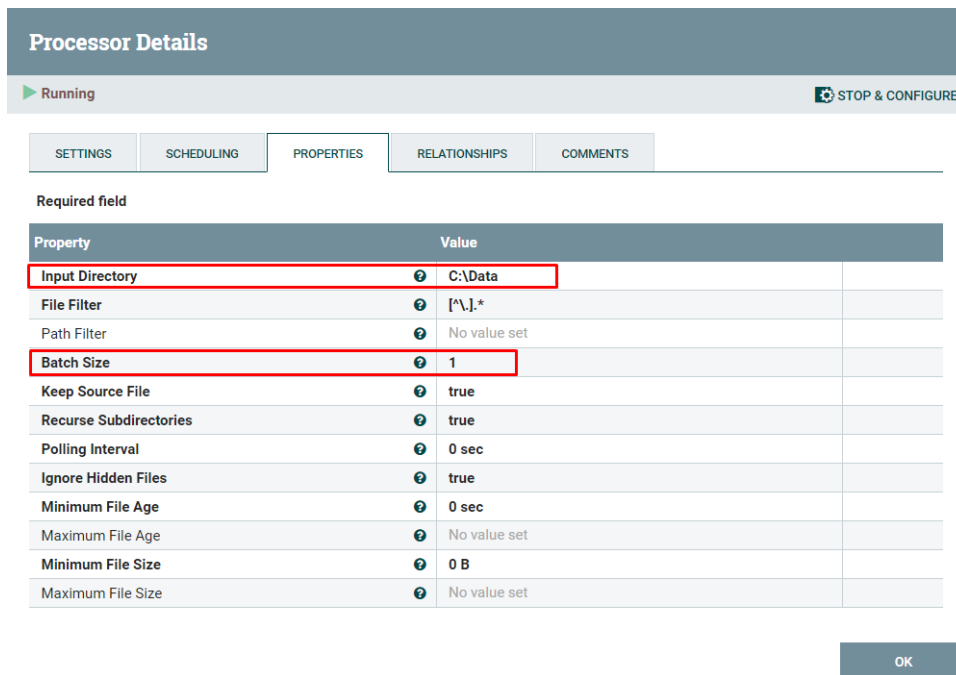
## 7.1.2. Kreiranje procesa "GetFile"

Na početku, postaviti ćemo u Apache NiFi prvi proces nazvan "GetFile", koji nam omogućuje učitavanje CSV datoteke. Kliknemo na ikonu procesora, pronađemo procesor "GetFile" i jednostavno ga dodamo pritiskom na tipku "Dodaj".



Slika 30: Procesor GetFile(Izvor: vlastita izrada)

Nakon što dodamo procesor, otvaramo ga dvostrukim klikom i postavljamo Scheduling interval na 20 sekundi. Zatim navodimo putanju do naše CSV datoteke. Preporučljivo je stvoriti određenu mapu na disku i tamo pohraniti datoteku (u mom slučaju, datoteka se nalazi u mapi "Data" na C disku). Također, konfiguriramo opciju "BatchSize" na 1, što određuje maksimalni broj datoteka koje se dohvaćaju u svakoj iteraciji i "KeepSourceFile" postavimo na True, kako nam se datoteka na izvoru ne bi obrisala.



Slika 31: Postavke procesora GetFile(Izvor: vlastita izrada)

Kako bismo provjerili je li podatke učitao Apache NiFi, stvorit ćemo novi proces pod nazivom "Wait", koji će služiti samo za tu svrhu. Kada mišem pređemo preko procesora "Getfile", pojavit će se strelica za uspostavljanje veze, a zatim jednostavno povucimo i isпустimo strelicu na procesor "Wait". Nakon što smo povezali procesore, možemo pokrenuti procesor "GetFile".

Kliknemo desnim gumbom miša na procesor "GetFile" i otvorimo DataProvenance kako bismo vidjeli podatke koji su učitani u Apache NiFi. Možemo preuzeti lokalnu kopiju učitanih podataka na naš disk ili kliknemo "View" za prikaz prikupljenih podataka.

**Provenance Event**

DETAILS ATTRIBUTES CONTENT

**Input Claim**

Container  
No value previously set

Section  
No value previously set

Identifier  
No value previously set

Offset  
No value previously set

Size  
No value previously set

**Output Claim**

Container  
default

Section  
587

Identifier  
1687945585433-29259

Offset  
1047

Size  
297.52 KB

DOWNLOAD VIEW

**Replay**

Cannot replay data from Provenance Event because the Source FlowFile Queue with ID No Value no longer exists

OK

Slika 32: Prikaz učitanih podataka(Izvor: vlastita izrada)

### 7.1.3. Kreiranje procesora "SplitText"

Procesor koji ćemo sada stvoriti je nazvan "SplitText". Ovaj procesor, kao što samo ime sugerira, ima funkciju razdvajanja tekstualne datoteke na više manjih tekstualnih datoteka prema granicama linija. Ovaj postupak može biti ograničen brojem linija ili ukupnom veličinom fragmenta. Na primjer, ako imamo CSV datoteku sa 2100 redaka, Apache NiFi će generirati 2100 manjih tekstualnih datoteka, pri čemu će svaki redak biti smješten u jednu od tih datoteka.

Kako bismo konfigurirali ovaj procesor, potrebno je postaviti određene opcije: "Line Split Count" i "Header Line Count". Opcija "Line Split Count" određuje broj linija koji će biti dodani u svakoj generiranoj datoteci. U našem slučaju, postaviti ćemo vrijednost na 1 jer svaki redak u CSV datoteci predstavlja jedan zaseban redak u bazi podataka. Također, opcija "Header Line Count" bit će postavljena na 1 jer prvi redak u CSV datoteci sadrži nazive atributa.

Na taj način, pomoću procesora "SplitText" možemo efikasno razdvojiti tekstualnu datoteku na manje fragmente, omogućujući nam daljnju obradu i manipulaciju podacima unutar Apache NiFi platforme.

**Processor Details**

▶ Running ⚙ STOP & CONFIGURE

SETTINGS | SCHEDULING | **PROPERTIES** | RELATIONSHIPS | COMMENTS

Required field

| Property                      | Value        |
|-------------------------------|--------------|
| Line Split Count              | 1            |
| Maximum Fragment Size         | No value set |
| Header Line Count             | 1            |
| Header Line Marker Characters | No value set |
| Remove Trailing Newlines      | true         |

OK

Slika 33: Postavke procesora SplitText(Izvor: vlastita izrada)

### 7.1.4. Kreiranje kontrolnih servisa

Kontrolni servisi u Apache NiFi-u su modularne komponente koje pružaju zajedničku funkcionalnost i resurse drugim komponentama unutar instance NiFi-a. Oni djeluju kao točke proširenja koje se mogu dodati, konfigurirati i koristiti od strane upravitelja protoka podataka (DFM) u korisničkom sučelju NiFi-a. Dodajemo 4 kontrolna servisa: CSVReader, JSONRecordSetWriter,DBCPCConnectionPool i AvroRecordSetWriter.

| Name                | Type                       | Bundle   | State   | Scope         |
|---------------------|----------------------------|--|---------|---------------|
| AvroRecordSetWriter | AvroRecordSetWriter 1.22.0 | org.apache.nifi-nfr-record-serialization-se... | Enabled | USGS CSV Data |
| CSVReader           | CSVReader 1.22.0           | org.apache.nifi-nfr-record-serialization-se... | Enabled | USGS CSV Data |
| DBCPCConnectionPool | DBCPCConnectionPool 1.22.0 | org.apache.nifi-nfr-dbcpc-service-nar          | Enabled | USGS CSV Data |
| JsonRecordSetWriter | JsonRecordSetWriter 1.22.0 | org.apache.nifi-nfr-record-serialization-se... | Enabled | USGS CSV Data |

Slika 34: Kontrolni servisi(Izvor: vlastita izrada)

Za dodavanje servisa, jednostavno desnom tipkom miša kliknemo na procesnu grupu i odaberemo opciju "Konfiguriraj". Nakon toga, otvara se prozor "ControllerService" gdje dodajemo prvi servis pod nazivom "CSVReader". Važno je napomenuti da ne

mijenjamo nijednu postavku unutar ovog koraka.

Sljedeći servis je "AvroRecordSetWriter" koji se koristi za pretvaranje ulaznih podataka iz CSV formata u Avro format. Avro format je format za serijalizaciju podataka koji se koristi za kompaktno pohranjivanje i razmjenu podataka. On pruža binarni format, specifikaciju sheme i bogat skup tipova podataka, čime je pogodan za obradu i integraciju velikih skupova podataka. Ovaj korak je dodan kako bismo prikazali primjer Avro formata unutar Apache NiFi platforme. Nema promjena u zadanim postavkama za ovaj servis.

Treći servis, "JSONRecordSetWriter", transformira ulazne podatke u JSON format za izlaz. Ovdje također zadržavamo zadane postavke servisa.

Kada želimo uspostaviti vezu s MySQL bazom podataka, koristimo DBCPConnectionPool. Prije konfiguracije, važno je preuzeti MySQL JDBC driver i uspješno ga instalirati. Nakon toga, možemo nastaviti s konfiguriranjem vrijednosti unutar samog servisa kako bismo ostvarili željenu vezu s bazom podataka.

Ovim koracima osiguravamo ispravno dodavanje servisa, pretvorbu podataka u željene formate i uspostavu veze s MySQL bazom podataka unutar Apache NiFi platforme.

| Property                     | Value  |
|------------------------------|--|
| Database Connection URL      | jdbc:mysql://127.0.0.1:3306/usgs             |
| Database Driver Class Name   | com.mysql.cj.jdbc.Driver                     |
| Database Driver Location(s)  | C:\Program Files (x86)\MySQL\Connector J 8.0 |
| Kerberos User Service        | No value set                                 |
| Kerberos Credentials Service | No value set                                 |
| Kerberos Principal           | No value set                                 |
| Kerberos Password            | No value set                                 |
| Database User                | Igor   |
| Password                     | Sensitive value set                          |
| Max Wait Time                | 500 millis                                   |
| Max Total Connections        | 8  |
| Validation query             | No value set                                 |
| Minimum Idle Connections     | 0  |
| Max Idle Connections         | 0  |

Slika 35: Postavke servisa „DBCPCConnectionPool“ (Izvor: vlastita izrada)

Za početak, potrebno je konfigurirati "Database Connection URL". Uobičajeni format URL-a je jdbc:mysql://localhost:port/nazivBaze. Nakon toga, nastavljamo s konfiguracijom "Database Driver Class Name", koji se obično navodi kao com.mysql.cj.jdbc.Driver. Također,

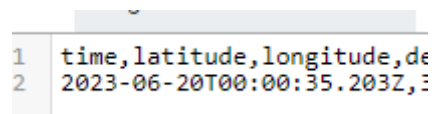
bitno je postaviti putanju do instalacije JDBC drivera, a ta putanja se često nalazi na lokaciji "C:\Program Files\MySQL\Connector".

Nakon što smo obavili sve navedene konfiguracije, ostaje nam još postaviti "Database User" i, ako je primjenjivo, lozinku. Ukoliko niste sigurni koje korisničko ime koristiti za pristup bazi podataka, možete jednostavno otvoriti MySQL i izvršiti naredbu "SELECT user()".

Ovom konfiguracijom osiguravamo uspostavu ispravne veze s bazom podataka i pravilno konfiguriramo sve potrebne parametre za neometano rukovanje podacima.

### 7.1.5. Kreiranje procesora "Update Record"

Sljedeći procesor koji planiramo upotrijebiti naziva se "Update Record". Ova komponenta ima važnu ulogu u izmjeni formata datuma unutar naše CSV datoteke. Prikaz formata datuma možete vidjeti na donjoj slici:

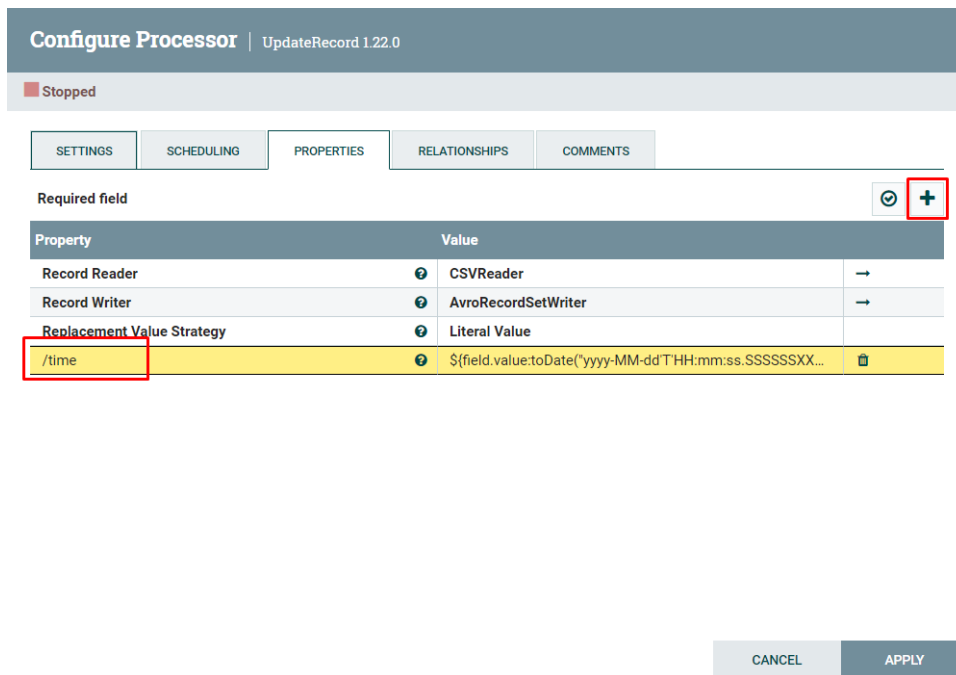


```
1 time, latitude, longitude, de
2 2023-06-20T00:00:35.203Z, :
```

Slika 36: Format datuma(Izvor: vlastita izrada)

Iz prikazanog formata na slici, želimo dobiti format u obliku 'yyy-MM-dd HH:mm:ss'. Kako bismo pravilno konfigurirali ovaj procesor, potrebno je postaviti ulazne podatke kao CSV podatke koje dobivamo iz prethodnog procesora, dok će izlazni podaci biti u Avro formatu.

Važno je naglasiti da atribut koji sadrži datum nosi naziv "Time". Da bismo izvršili promjenu ovog atributa, otvaramo konfiguraciju procesora "Update Record" te dodajemo novo svojstvo pod nazivom /time. Novo svojstvo možemo jednostavno dodati pritiskom na "+" i upisivanjem naziva svojstva u odgovarajuće polje. Ovaj procesor pruža fleksibilnost i mogućnost precizne manipulacije podacima, omogućavajući nam da prilagodimo format datuma prema našim specifičnim zahtjevima.

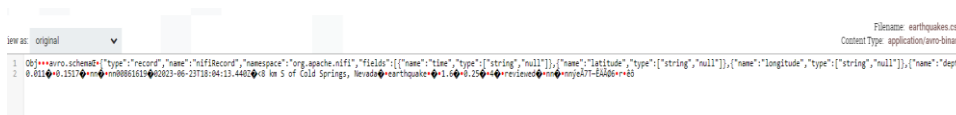


Slika 37: Postavljanje novog formata datuma(Izvor: vlastita izrada)

Nakon dodavanja novog svojstva, primjenjujemo funkciju "toDate" kako bismo promijenili format iz naše CSV datoteke. Cjelokupna naredba izgleda ovako:

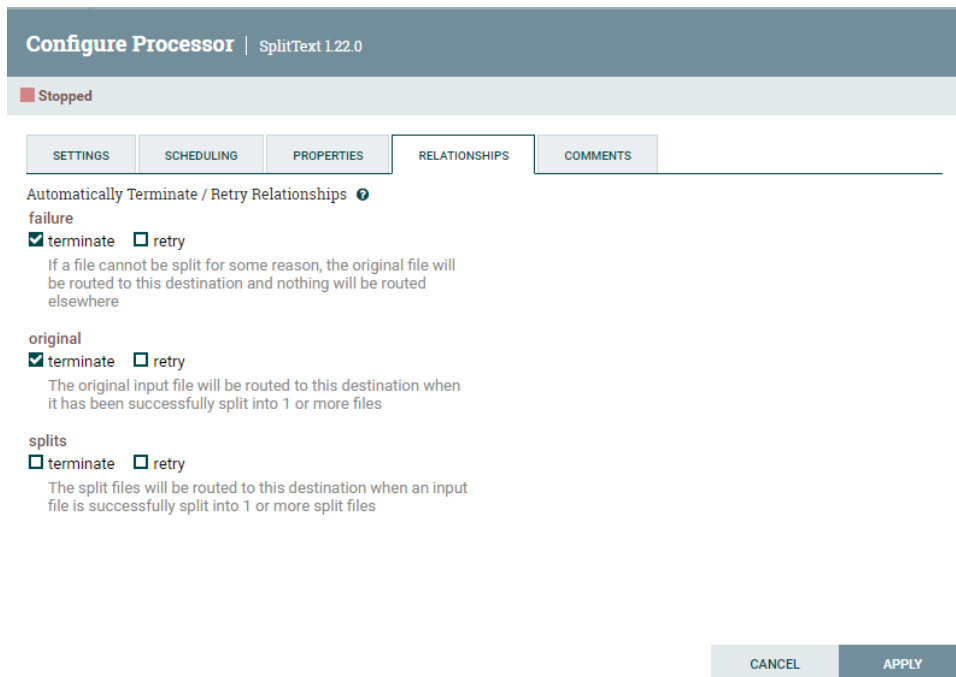
```
$(field.value.toDate("yyyy-MM-dd'T'HH:mm:ss.SSSSSSXXX")):format("yyyy-MM-dd HH:mm:ss.SSS", '+00:00')
```

Također, primjer Avro formata možemo vidjeti na sljedećoj slici.



Slika 38: Prikaz Avro formata(Izvor: vlastita izrada)

Kada vršimo povezivanje između procesora, neophodno je prilagoditi postavke veze. U ovom konkretnom slučaju, uspostavljamo vezu između procesora pod nazivom "SplitText" i "UpdateRecord". Naš glavni cilj je preusmjeriti sve generirane podijeljene datoteke ka novom procesoru, dok originalna datoteka ne nastavlja dalje. Ukoliko neki podaci ne uspiju biti preusmjereni, ti procesi će biti završeni.

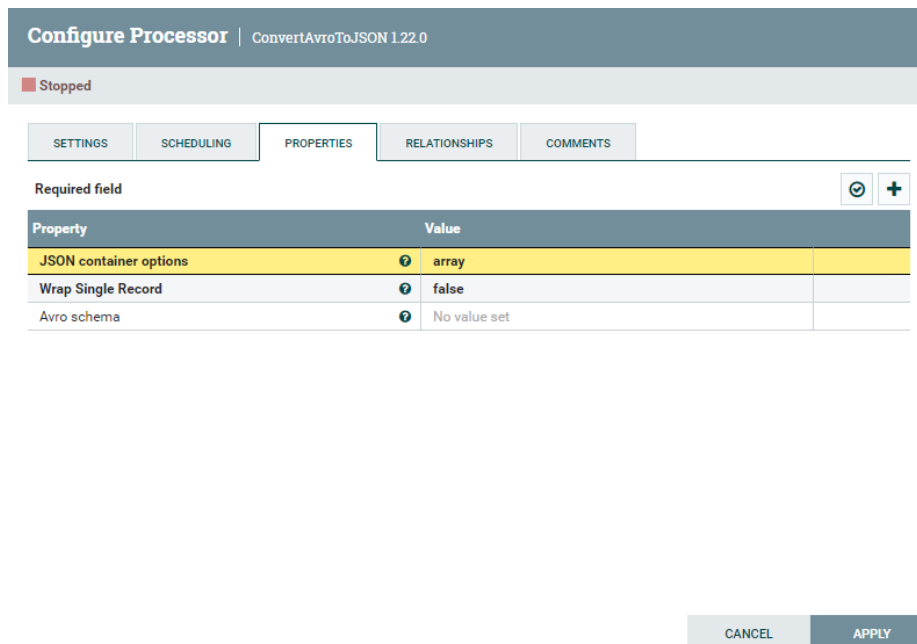


Slika 39: Postavke veze između procesora(Izvor: vlastita izrada)

### 7.1.6. Kreiranje procesora ConvertAvroToJSON

Budući da su izlazni podaci iz prethodnog procesora već bili u Avro formatu, naš sljedeći korak je konverzija tih podataka u JSON format radi spremanja u MySQL bazu podataka. U tu svrhu koristimo poseban procesor koji je namijenjen za tu konverziju. Važno je napomenuti da u ovom procesoru nećemo mijenjati nijednu postavku jer želimo zadržati zadane vrijednosti koje su prikladne za našu svrhu. Kroz ovaj postupak osiguravamo da naši podaci budu pravilno obrađeni i spremni za daljnju upotrebu u bazi podataka.

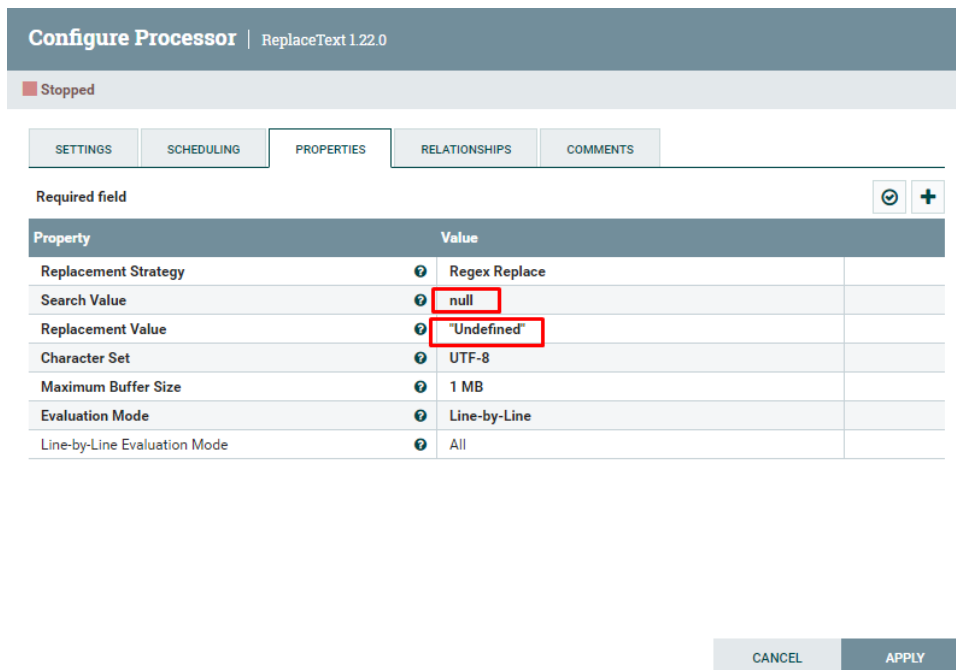




Slika 40: Postavke procesora ConvertAvroToJSON(Izvor: vlastita izrada)

### 7.1.7. Kreiranje procesora "ReplaceText"

Sljedeći korak u našem postupku je stvaranje procesora poznatog kao "ReplaceText". Ova funkcionalnost je od iznimne važnosti jer nam omogućuje zamjenu specifičnih vrijednosti unutar datoteke. U konkretnom primjeru koji pratimo, vršimo transformaciju koja se sastoji od zamjene svakog atributa koji sadrži vrijednost "NULL" s izrazom "Undefined". Kroz ovu jednostavnu, ali moćnu operaciju, osiguravamo da naši podaci budu točni, čitljivi i konzistentni.

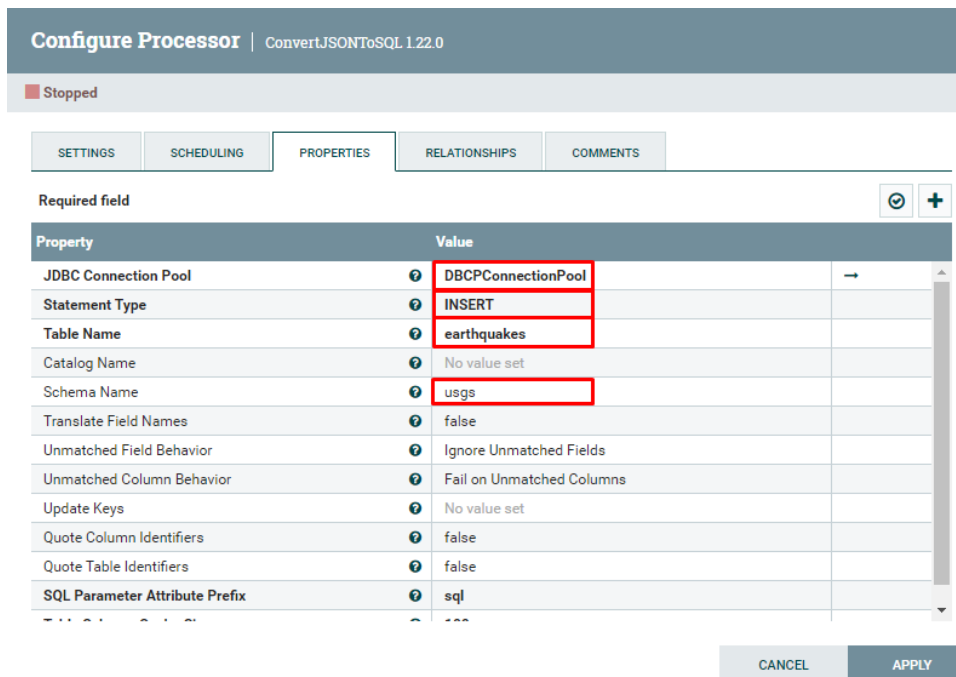


Slika 41: Postavke procesora „ReplaceText“ (Izvor: vlastita izrada)

Pri konfiguraciji samog procesora, potrebno je postaviti "SearchValue" na null kako bismo pronašli sve vrijednosti koje su NULL, a zatim "ReplacementValue" postaviti na vrijednost kojom želimo zamijeniti (u ovom slučaju, postavljam na "Nedefinirano"). U praksi se često koristi zamjena NULL vrijednosti s -1 ili drugim definiranim brojem od strane korisnika, no za potrebe ovog rada koristimo "Nedefinirano".

### 7.1.8. Kreiranje procesora “ConvertJSONToSQL”

Sljedeći korak u ovom procesu je stvaranje procesora pod nazivom "ConvertJSONToSQL." Glavna funkcija ovog procesora je pretvoriti FlowFile, prethodno pretvoren iz Avro formata u JSON, u SQL naredbu za izvođenje operacija UPDATE, INSERT ili DELETE. U našem konkretnom primjeru koristimo naredbu INSERT za prikaz podataka unesenih u našu bazu podataka..



Slika 42: Postavke procesora „ConvertJSONToSQL“ (Izvor: vlastita izrada)

Unutar konfiguracije samog procesora, prvi korak je definiranje prethodno kreiranog servisa poznatog kao "DBCPCConnectionPool". Taj korak je bitan kako bi procesor bio usmjeren prema ispravnom serveru i odgovarajućoj bazi podataka.

Sljedeći korak je odabir "StatementType" parametra, koji nam omogućuje odabir vrste SQL naredbe koju želimo izvršiti. Imamo mogućnost odabrati između INSERT, UPDATE ili DELETE. Osim toga, potrebno je navesti ime tablice unutar same baze podataka. U našem slučaju, ta tablica nosi naziv "earthquakes". Konačno, važno je specificirati i ime sheme (schema name) kako bi procesor znao u kojoj shemi je definirana navedena tablica.

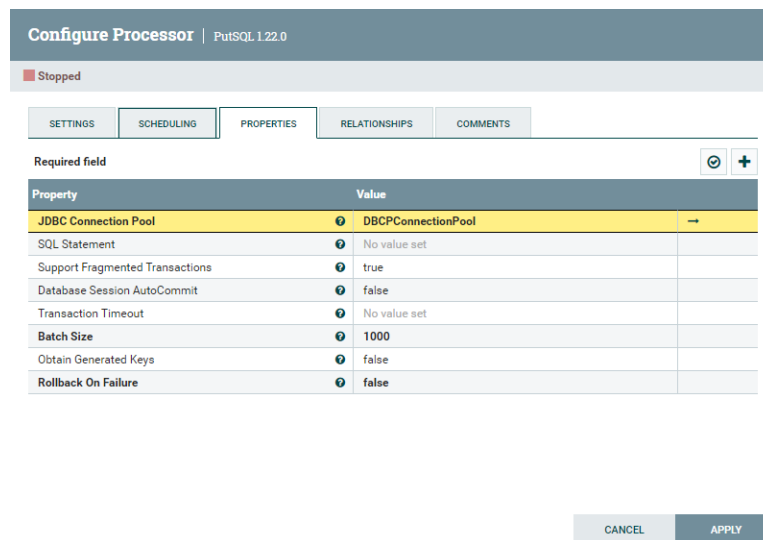
### 7.1.9. Kreiranje procesora PutSQL

Procesor PutSql u Apache NiFi platformi omogućuje integraciju i upis podataka u bazu podataka putem SQL upita. Ovaj procesor preuzima podatke iz dolaznog toka i izvršava SQL upit za njihovu pohranu u određenu bazu podataka. PutSql procesor podržava različite vrste baza podataka i omogućuje konfiguriranje parametara kao što su veza s bazom podataka, tablica za upisivanje podataka i SQL upit za izvršavanje.

Konfiguracija procesora PutSql je prilično jednostavna. Dodajemo ga na radnu površinu i samo povezujemo JDBC Connection Pool s prethodno konfiguriranim servisom

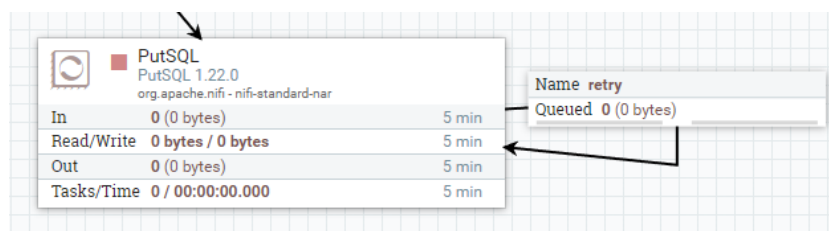
"DBCPConnectionPool".

Dodatna opcija koju imamo je mogućnost ručnog unosa SQL izjava. Na primjer, ako želimo izvršiti ažuriranje nad podacima koji prolaze kroz procesor, možemo ručno upisati SQL upit za promjenu određenih podataka. Ova funkcionalnost omogućuje nam veću kontrolu nad izvršavanjem SQL operacija i prilagođavanje procesora našim specifičnim potrebama.



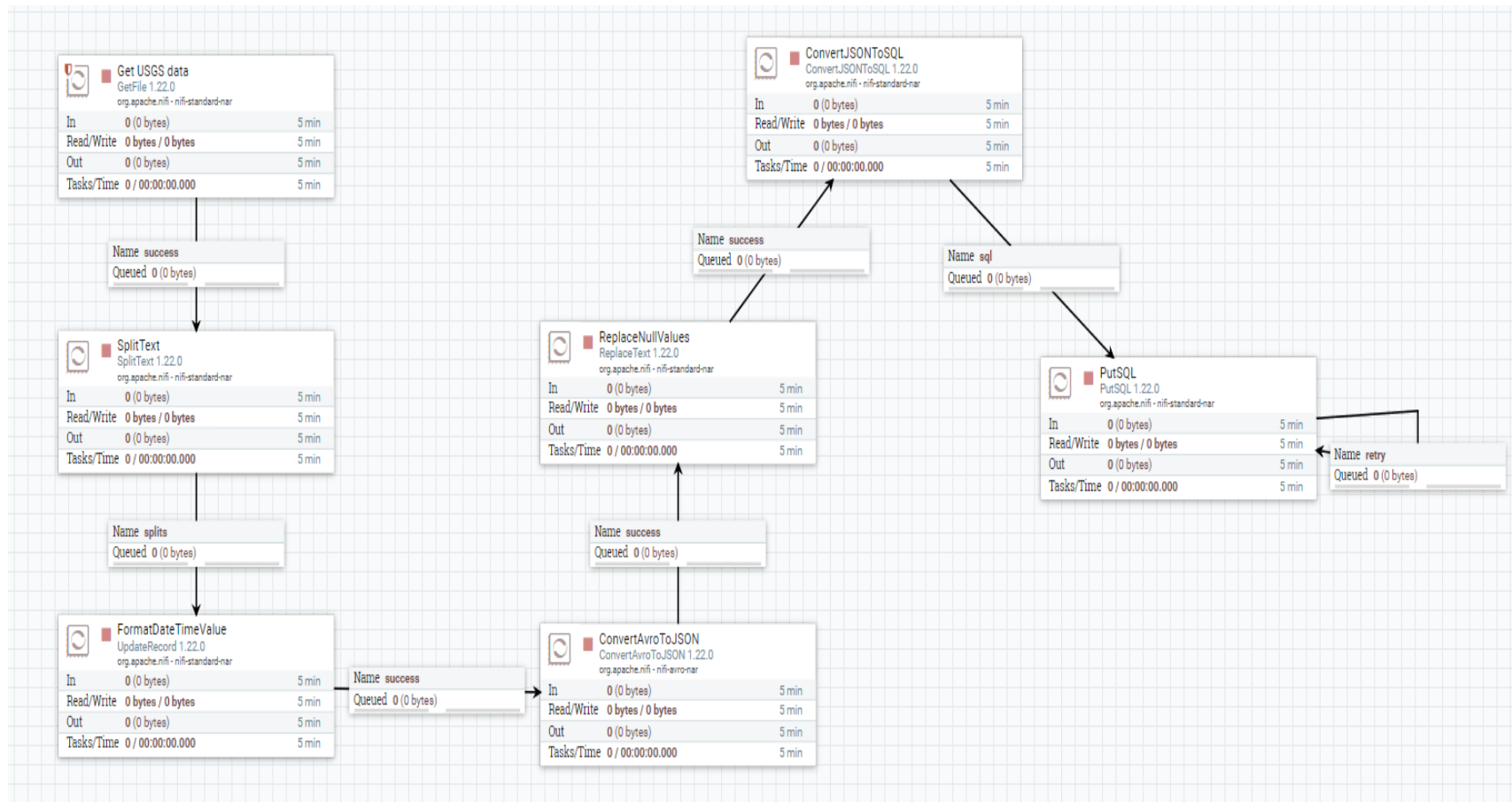
Slika 43: Postavke procesora PutSQL(Izvor: vlastita izrada)

Nadalje, nakon što smo prethodno podijelili datoteku na više dijelova pomoću procesora SplitText, output samog procesora "PutSql" je zapravo usmjeren natrag na sebe. Postavljamo povezivanje na vlastiti procesor i uključujemo opciju Retry kako bi se za svaki od tih dijelova izvršio odgovarajući SQL upit, u ovom slučaju upisivanje podataka u bazu.



Slika 47: Prikaz veze procesora PutSQL(Izvor: vlastita izrada)

Kada smo uspješno konfigurirali i povezali sve procesore, na donjoj slici možemo vidjeti cjelokupni prikaz dataflowa za prijenos podataka iz CSV datoteke u MySQL bazu podataka.



Slika 44: Prikaz finalnog dataflowa(Izvor: vlastita izrada)

## 8. Apache NiFi i Apache Airflow

Apache Airflow i Apache NiFi su kao dvije različite pjesme koje sviraju na nešto drugačijem tonu. Iako su suštinski različiti, i Apache Airflow i Apache NiFi alati su oblikovani za upravljanje neprocjenjivom imovinom većine organizacija: podacima.

### 8.1. Osnovne karakteristike



Slika 45: Apache NiFi vs Apache Airflow (astronomer.io, dostupno 2023.)

Priroda Airflowa je usmjerena prema orkestraciji, dok NiFi prvenstveno cilja na automatizaciju prijenosa podataka između sustava. Airflow se može smatrati "Upraviteljem radnih tijekova", dok Apache NiFi spada u kategoriju "Obrade u stvarnom vremenu". Ovi alati nisu međusobno isključivi, već nude uzbudljive mogućnosti i pomažu u prevladavanju prepreka vezanih uz podatkovne silose. (Wrzosińska, 2023)

S obzirom na kontinuirani rast količine podataka, organizacije sve više traže projekte za pohranu podataka i napredna analitička rješenja. ETL (Extract, Transform, Load) je ključna komponenta modernog podatkovnog ekosustava jer osigurava uspješnu integraciju podataka u različite baze podataka i aplikacije. I Airflow i NiFi su vrhunski ETL alati. Da biste odabrali pravi alat za svoje potrebe, važno je razmotriti što točno želite postići sa svojim podacima.

## 8.2. Različitosti

Različitosti između Apache NiFi i Apache Airflowa bit će objašnjene u donjoj tablici:

| Apache Airflow  | Apache NiFi   |
|---|---|
| Besplatni Python alat za automatizaciju tijekom rada otvorenog koda koji može stvarati i upravljati složenim cjevovodima podataka.  | Apache NiFi je ETL alat s programiranjem temeljenim na protoku koji uključuje web sučelje koje čini rukovanje protokom podataka jednostavnim (povuci i ispusti).  |
| Za stvaranje ETL cjevovoda, možemo kombinirati operatore prijenosa Airflow s operatorima baze podataka. Moramo koristiti probnu pohranu Google Cloud jer ne postoji Airflow operator za izravno premještanje podataka iz Postgresa u BigQuery.  | Procesori u NiFi radnom tijeku mogu obrađivati, potvrđivati, filtrirati, spajati, dijeliti ili mijenjati podatke. FlowFile Controller pomaže u upravljanju resursima između tih komponenti dok prenose podatke u obliku FlowFiles preko povezanih redova čekanja. |
| Korisničko sučelje Airflowa prilično je jednostavno i nudi izvrstan način za dohvat metapodataka. Možemo jednostavno uključiti i isključiti rasporede, vidjeti status DAG-a, pokrenuti SQL upite, pratiti cjevovode koji se koriste, držati ih na oku i brzo popraviti sve nove probleme. | Postoje prednosti i mane jednostavnog korisničkog sučelja Apache NiFi. Nije izvanredan, ali je jednostavan, bazičan i čist bez dodatnih elemenata. NiFi nudi web-bazirano korisničko sučelje koje je prilično fleksibilno   |
| Apache Airflow je popularniji jer ima preko 27 tisuća Github zvijezda i 21 tisuća suradnika.  | Apache NiFi postupno postaje sve popularniji s 3,2 tisuće Github zvijezda i 400 suradnika.  |
| Korisnicima omogućuje izgradnju podatkovnih cjevovoda s osnovnim značajkama Pythona kao što su petlje za generiranje zadataka i formati vremena podataka za raspoređivanje. Omogućuje korisnicima da što dinamičnije stvaraju podatkovne kanale.  | Apache NiFi nudi modul podrijetla podataka za praćenje i nadzor podataka od početka toka do kraja. Programeri mogu dizajnirati svoje prilagođene aktivnosti izvješćivanja i procesore koji odgovaraju njihovim zahtjevima   |
| Apache Airflow koristi senzore za preuzimanje kontrole nad izvršavanjem zadatka ako nedostaju određeni uvjeti.  | Uz podršku upravljanja korisničkim ulogama, Apache NiFi može koristiti LDAP za autorizaciju.  |

Tablica 3: Apache NiFi vs Apache Airflow

Ono što Airflow i NiFi zasigurno imaju zajedničko je to što su alati otvorenog koda i temeljeni na zajednici. Čini se da Airflow ima šire odobrenje s 27.2K GitHub zvijezda i 9.2k forkova te više suradnika. Vjerojatno je to zbog činjenice da ima više primjena, jer po prirodi Airflow služi u različite svrhe od NiFi-a. Ipak, oba alata mogu ponuditi puno ugrađenih operatora, stalna ažuriranja i podršku svojih zajednica.

NiFi je savršen alat za rukovanje velikim podacima - njihovo izdvajanje i učitavanje u zadani prostor. To je proširiva platforma poznata po izvrsnom rukovanju pogreškama i jednostavnom sučelju. Ne postoji bolji izbor kada je riječ o "postavi i zaboravi" vrsti cjevovoda, jer NiFi ne nudi praćenje uživo niti statistiku po zapisu. Savršen je ako se uopće ne želite baviti kodiranjem - NiFi se temelji na "povuci i ispusti", a ovaj je alat sigurno "go-to" rješenje za grupno strujanje uživo. Mogućnosti raspoređivanja nisu jako robusne, ali tehnički, NiFi nije raspoređivač.

S druge strane, Airflow je savršeno rješenje za raspoređivanje određenih zadataka, postavljanje ovisnosti i upravljanje programskim tijekovima rada. Velika, živahna zajednica stalno ažurira alat i čini ga boljim sa svakim ažuriranjem. Airflow uklanja mnogo posla iz ruku vašeg IT tima, jer je to jedna od najrobusnijih platformi za orkestriranje radnih tokova ili cjevovoda. Omogućuje da jednostavno vidite ovisnosti, kod, zadatke okidača, napredak, zapisnike i status uspjeha vaših podatkovnih cjevovoda. Airflow je orkestrator podataka koji nadilazi upravljanje podacima - pomaže u pružanju uvida temeljenih na podacima, što rezultira rastom poduzeća.

“Prije Airflowa naši su cjevovodi bili podijeljeni, neke stvari su rađene na Cronu, neke na NiFi-ju, neke na drugim alatima. Htjeli smo sve to spojiti. Uz NiFi, programeri CRED-a morali su napraviti kopiju svih cjevovoda i koristiti ih za svoje specifične slučajeve upotrebe. Ako su željeli nešto promijeniti, na primjer, nadograditi na noviju verziju Airflowa, morali su ažurirati obje kopije,” rekao je Omesh Patil, Data Architect u CRED-u.

Ukratko, ne postoji "bolji" alat. Sve ovisi o točnim potrebama - NiFi je savršen za osnovni ETL proces velikih podataka, dok je Airflow "go-to" alat za planiranje i izvršavanje složenih radnih procesa, kao i poslovno kritičnih procesa.



## 9. Prednosti nad tradicionalnim alatima i nedostaci

### Apache NiFi prednosti

- **Jednostavnost korištenja**

Apache NiFi pruža intuitivno i jednostavno grafičko sučelje za dizajniranje i upravljanje protokom podataka. Ovaj vizualni pristup pojednostavljuje stvaranje i modificiranje složenih radnih procesa integracije.

- **Izvor podataka**

NiFi pruža ugrađeno praćenje izvora podataka, omogućavajući vidljivost od početka do kraja i reviziju protoka podataka. Ova značajka omogućuje korisnicima praćenje izvora, transformacije i odredišta podataka, što poboljšava upravljanje podacima, usklađenost i mogućnosti rješavanja problema.

- **Skalabilnost**

Apache NiFi dizajniran je za horizontalno skaliranje, što mu omogućuje rukovanje velikim količinama podataka i prilagođavanje rastućim potrebama obrade podataka.

- **Sigurnost podataka**

NiFi nudi jake sigurnosne značajke, uključujući kontrolu pristupa temeljenu na ulogama, SSL/TLS enkripciju i enkripciju u mirovanju.

- **Fleksibilnost i proširivost**

NiFi podržava razvoj i integraciju prilagođenih procesora, dopuštajući korisnicima da prošire njihovu funkcionalnost kako bi zadovoljili specifične potrebe. Također sadrži bogat ekosustav procesora koje je osigurala zajednica, omogućujući korisnicima da iskoriste širok raspon unaprijed izgrađenih komponenti za različite računalne zadatke.

### Apache NiFi nedostaci

- **Krivulja učenja**

Vizualno sučelje i jedinstveni model protoka podataka Apache NiFi-a mogu zahtijevati određeno učenje i prilagodbu za korisnike koji su navikli na tradicionalne pristupe temeljene na kodiranju.

- **Ograničene mogućnosti skriptiranja:** Za razliku od tradicionalnih alata koji pružaju opsežne mogućnosti skriptiranja, Apache NiFi se prvenstveno oslanja na vizualno sučelje za dizajniranje protoka podataka.
- **Preopterećenje performansi**

Grafičko sučelje Apache NiFi-a i praćenje porijekla podataka uzrokuju određeno opterećenje performansama u usporedbi s laganim alatima koji se temelje na skripti. U scenarijima u kojima je iznimno niska latencija ili obrada visokih performansi kritična, tradicionalni alati s minimalnim opterećenjem mogli bi biti prikladniji.
- **Zahtjevi za resurse**

Zbog svoje arhitekture temeljene na Javi i prirode bogate značajkama, Apache NiFi može zahtijevati više sistemskih resursa (CPU, memorija, prostor na disku) u usporedbi s laganim tradicionalnim alatima
- **Zajednica i ekosustav**

Iako Apache NiFi ima aktivnu zajednicu i rastući ekosustav procesora, konektora i proširenja, još uvijek može imati relativno manju korisničku bazu u usporedbi s nekim dugotrajnim tradicionalnim alatima.
- **Integracija naslijeđenog sustava**

Tradicionalni alati mogu imati bolju kompatibilnost i mogućnosti integracije s naslijeđenim sustavima koji su prevladavali prethodnih godina. Apache NiFi, budući da je noviji alat, može zahtijevati dodatne napore ili prilagođeni razvoj za besprijekornu integraciju sa starijim, vlasničkim sustavima.

## 10. Zaključak

U ovom radu provedena je detaljna analiza i studija integracije podataka pomoću Apache NiFi alata. Cilj rada bio je razumjeti mogućnosti i prednosti Apache NiFi u integraciji podataka te ispitati njegovu primjenu u stvarnom svijetu.

Analizirali smo značajke i mogućnosti Apache NiFi-ja, uključujući intuitivno korisničko sučelje, podršku za različite izvore podataka, mogućnosti fleksibilne obrade, upravljanje sigurnošću i više. Na temelju istraživanja i razvoja praktičnog dijela rada preporuča se korištenje Apache NiFi za integraciju podataka.

Alat je idealan za scenarije koji zahtijevaju obradu podataka u stvarnom vremenu, npr. analiza u stvarnom vremenu, protok podataka i sinkronizacija različitih izvora podataka. Kao rezultat ovog rada, istaknuli smo važnost i prednosti Apache NiFi u integraciji podataka.

Ova tehnologija može tvrtkama dati konkurentsku prednost pojednostavljivanjem protoka podataka i omogućavanjem brzih, informiranih odluka temeljenih na ažurnim podacima.

## Popis literature

[1] nifi.apache.org Apache NiFi Overview

<https://nifi.apache.org/docs.html>, dostupno 30.6.2023.

[2] intelligentpathways.com, Using Apache NiFi for Enterprise Workflow Automation

<https://intelligentpathways.com.au/using-apache-nifi-for-enterprise-workflow-automation/> , dostupno 30.6.2023.

[3] learningjournal.guru, What is NiFi?

<https://www.learningjournal.guru/article/apache-nifi/what-is-nifi/>, dostupno 30.6.2023.

[4] docs.vmware.com, Using the Apache NiFi User Interface

[https://docs.vmware.com/en/VMware-Greenplum-Connector-for-Apache-NiFi/1.1/greenplum-connector-nifi/using\\_nifi\\_ui.html](https://docs.vmware.com/en/VMware-Greenplum-Connector-for-Apache-NiFi/1.1/greenplum-connector-nifi/using_nifi_ui.html) , dostupno 30.6.2023.

[5] tutorialspoint.com, Apache NiFi – Introduction

[https://www.tutorialspoint.com/apache\\_nifi/apache\\_nifi\\_introduction.htm](https://www.tutorialspoint.com/apache_nifi/apache_nifi_introduction.htm) , dostupno 25.6.2023.

[6] digitalis.io, What is Apache NiFi?

<https://digitalis.io/blog/dataops/what-is-apache-nifi/> , dostupno 25.6.2023.

[7] javatpoint.com, Apache NiFi

<https://www.javatpoint.com/apache-nifi> , dostupno 30.6.2023.

[8] pierrevillard.com, Transform data with Apache NiFi

<https://pierrevillard.com/2016/03/09/transform-data-with-apache-nifi/>, dostupno 30.6.2023.

[9] docs.cloudera.com, Apache NiFi Expression Language Guide

<https://docs.cloudera.com/HDPDocuments/HDF3/HDF-3.5.2/expression-language-guide/hdf-expression-language-guide.pdf> , dostupno 30.6.2023.

[10] projectpro.io, Apache NiFi vs Airflow- Which ETL Tool is Better?

<https://www.projectpro.io/article/apache-nifi-vs-airflow/617>, dostupno 30.6.2023.

- [11] cdata.com, Bridge MySQL Connectivity with Apache NiFi  
<https://www.cdata.com/kb/tech/mysql-jdbc-apache-nifi.rst>, dostupno 30.6.2023.
- [12] czasopisma.mazowiecka.edu.p, APACHE NIFI AS A TOOL FOR STREAM PROCESSING OF MEASUREMENT DATA  
<https://czasopisma.mazowiecka.edu.pl/index.php/ne/article/view/901>, dostupno 30.6.2023.
- [13] revistaie.ase.ro, Big Data Analytics: Analysis of Features and Performance of Big Data Ingestion Tools  
<http://revistaie.ase.ro/content/86/03%20-%20matacuta,%20popa.pdf>, dostupno 30.6.2023.
- [14] astronomer.io, Apache NiFi vs Apache Airflow,  
<https://www.astronomer.io/blog/apache-nifi-vs-airflow/>, dostupno 30.6.2023.
- [15] medium.com, Apache NiFi in a nutshell,  
<https://vishnu-g.medium.com/apache-nifi-in-a-nutshell-ab63b2451891>, dostupno 30.06.2023.
- [15] docs.cloudera.com, Using Data Flow Provenance Tools,  
[https://docs.cloudera.com/HDPDocuments/HDF3/HDF-3.5.1/using-dataflow-provenance-tools/content/data\\_provenance.html](https://docs.cloudera.com/HDPDocuments/HDF3/HDF-3.5.1/using-dataflow-provenance-tools/content/data_provenance.html), dostupno 30.6.2023.
- [16] J. Hurwitz, A. Nugent, F. Halper, M. Kaufman, Big Data For Dummies, John Wiley & Sons, 2013.  
<https://jan.newmarch.name/loT/BigData/Big%20Data%20For%20Dummies.pdf>, dostupno 11.7.2023.
- [17] O'Reilly Team, Big Data Now: 2015 Edition, O'Reilly Media, 2015.  
[https://www.newLaw.gr/media/cms\\_categories/1/files/Big\\_Data\\_Now\\_Current\\_Perspectives\\_from\\_O.pdf](https://www.newlaw.gr/media/cms_categories/1/files/Big_Data_Now_Current_Perspectives_from_O.pdf), dostupno 12.7.2023.
- [18] For Big Data Analytics There's No Such Thing as Too Big The Compelling Economics and Technology of Big Data Computing March 2012 By: 4syth.com emerging big data thought leaders

<https://www.ilkogretim-online.org/fulltext/218-1662791122.pdf>, dostupno 12.7.2023.

[19] Divyakant Agrawal, UC Santa Barba, Philip Bernstein, Microsoft Elisa Bertino,...(2012), " Challenges and Opportunities with Big Data"

<https://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=1000&context=cctech>, dostupno 12.7.2023.

[20] InterviewBit.com, Big Data Architecture – Detailed Explanation,

<https://www.interviewbit.com/blog/big-data-architecture/>, dostupno 17.7.2023.

[21] Databricks.com, What is Big Data analytics?

<https://www.interviewbit.com/blog/big-data-architecture/>, dostupno 17.7.2023

[22] Streamsets.com, Understanding the Lambda Architecture,

<https://streamsets.com/blog/understanding-the-lambda-architecture/>, dostupno 17.7.2023.

[23] Heavy.ai, Big data architecture

<https://www.heavy.ai/technical-glossary/big-data-architecture>, dostupno 19.7.2023.

# Popis slika

Popis slika treba biti izrađen po uzoru na indeksirani sadržaj, te upućivati na broj stranice na kojoj se slika može pronaći.

|  |    |
|--|----|
| Slika 1: 3V karakteristike (Izvor: techtarget.com, 2023.)                      | 3  |
| Slika 2: Big data arhitektura(Izvor: InterviewBit.com, 2023.)                  | 4  |
| Slika 3: Lambda arhitektura(Izvor: streamsets.com, 2023.)                      | 5  |
| Slika 4: Kappa arhitektura(Izvor: databricks.com, 2023.)                       | 6  |
| Slika 5: NO-SQL baze podataka(Izvor: InterviewBit.com, 2023.)                  | 8  |
| Slika 6: Logo ASF-a (Izvor: apache.org, dostupno 2023.)                        | 12 |
| Slika 7: Tijek podataka (learningjournal.guru, dostupno 2023.)                 | 14 |
| Slika 8: Praćenje podataka preko tablice (Izvor: cloudera.com, dostupno 2023.) | 17 |
| Slika 9: Praćenje podataka pomoću grafa (Izvor: vlastita izrada)               | 18 |
| Slika 10: NiFi izražajni jezik (Izvor: apache.org, dostupno 2023.)             | 19 |
| Slika 11: NiFi izražajni jezik (Izvor: apache.org, dostupno 2023.)             | 20 |
| Slika 12: Apache NiFi arhitektura (digitalis.io, dostupno 2023.)               | 24 |
| Slika 13: NiFi arhitektura (digitalis.io, dostupno 2023.)                      | 26 |
| Slika 14: Apache NiFi komponente (Izvor: docs.vmware.com, dostupno 2023.)      | 26 |
| Sljedeća tablica prikazuje svaku komponentu i njihovu glavnu svrhu.            | 26 |
| Slika 15: NiFi procesori (Izvor: docs.vmware.com, dostupno 2023.)              | 27 |
| Slika 16 : Prikaz login identifikatora(Izvor: vlastita izrada)                 | 30 |
| Slika 17: LDAP arhitektura (vanducng.dev, dostupno 2023.)                      | 31 |
| Slika 18: Login-identity-providers.xml(Izvor: vlastita izrada)                 | 32 |
| Slika 19: Prikaz procesora iz perspektive Korisnika 2 (izvor: vlastita izrada) | 33 |
| Slika 20 : Prikaz postavljanja politike(izvor: vlastita izrada)                | 34 |
| Slika 21: Override politike(izvor: vlastita izrada)                            | 35 |
| Slika 23: Ikona komponente procesora(Izvor: vlastita izrada)                   | 37 |
| Slika 24: Dijaloški okvir procesora(Izvor: vlastita izrada)                    | 37 |
| Slika 25: Komponenta procesora(Izvor: vlastita izrada)                         | 38 |
| Slika 26: Povezivanje komponenta(Izvor: vlastita izrada)                       | 38 |
| Slika 27: Konfiguracija veze(Izvor: vlastita izrada)                           | 39 |
| Slika 28: Prikaz dataflow-a(Izvor: vlastita izrada)                            | 39 |
| Slika 29: Dodavanje procesne grupe(Izvor: vlastita izrada)                     | 41 |
| Slika 30: Procesor GetFile(Izvor: vlastita izrada)                             | 42 |
| Slika 31: Postavke procesora GetFile(Izvor: vlastita izrada)                   | 43 |
| Slika 32: Prikaz učitanih podataka(Izvor: vlastita izrada)                     | 44 |

|  |    |
|--|----|
| Slika 33: Postavke procesora SplitText(Izvor: vlastita izrada) .....           | 45 |
| Slika 34: Kontrolni servisi(Izvor: vlastita izrada) .....                      | 45 |
| Slika 35: Postavke servisa „DBCPCConnectionPool“ (Izvor: vlastita izrada)..... | 46 |
| Slika 36: Format datuma(Izvor: vlastita izrada).....                           | 47 |
| Slika 37: Postavljanje novog formata datuma(Izvor: vlastita izrada) .....      | 48 |
| Slika 38: Prikaz Avro formata(Izvor: vlastita izrada) .....                    | 48 |
| Slika 39: Postavke veze između procesora(Izvor: vlastita izrada) .....         | 49 |
| Slika 40: Postavke procesora ConvertAvrtoToJson(Izvor: vlastita izrada).....   | 50 |
| Slika 41: Postavke procesora „ReplaceText“ (Izvor: vlastita izrada) .....      | 51 |
| Slika 42: Postavke procesora „ConvertJSONToSQL“ (Izvor: vlastita izrada) ..... | 52 |
| Slika 43: Postavke procesora PutSQL(Izvor: vlastita izrada) .....              | 53 |
| Slika 47: Prikaz veze procesora PutSQL(Izvor: vlastita izrada) .....           | 53 |
| Slika 44: Prikaz finalnog dataflowa(Izvor: vlastita izrada).....               | 54 |
| Slika 45: Apache NiFi vs Apache Airflow (astronomer.io, dostupno 2023.) .....  | 55 |
| Slika 46: JDK download stranica(Izvor: vlastita izrada).....                   | 68 |
| Slika 47: Dodavanje varijable okruženja(Izvor: vlastita izrada) .....          | 69 |
| Slika 48: Dodavanje systemske varijable(Izvor: vlastita izrada) .....          | 69 |
| Slika 49: Prikaz nakon dodavanja varijable(Izvor: vlastita izrada).....        | 70 |
| Slika 50: Provjera instalacije JDK-a(Izvor: vlastita izrada).....              | 71 |
| Slika 51: Stranica za preuzimanje Nifi-a(Izvor: vlastita izrada) .....         | 72 |
| Slika 52: Pokretanje Nifi-a(Izvor: vlastita izrada) .....                      | 73 |
| Slika 53: Prikaz nakon pokretanja Nifi-a(Izvor: vlastita izrada).....          | 73 |
| Slika 54:Provjera porta(Izvor: vlastita izrada) .....                          | 74 |
| Slika 55: Početni zaslon Nifi-a(Izvor: vlastita izrada) .....                  | 75 |
| Slika 56: Sučelje Nifi-a(Izvor: vlastita izrada).....                          | 75 |
| Slika 57: Prikaz alatnih traka i postavka(Izvor: vlastita izrada).....         | 76 |
| Slika 58:Globalni izbornik(Izvor: vlastita izrada).....                        | 76 |



# Popis tablica

Popis tablica treba biti izrađen po uzoru na indeksirani sadržaj, te upućivati na broj stranice na kojoj se tablica može pronaći.

|  |    |
|--|----|
| Tablica 1: Povijest Apache NiFi-a (Izvor: javatpoint.com,dostupno 2023.) ..... | 13 |
| Tablica 2: Komponente Apache NiFi-a .....                                      | 27 |
| Tablica 3: Apache NiFi vs Apache Airflow .....                                 | 56 |

# Prilog

## 11. Instalacija

Dva su glavna koraka koja moramo izvršiti, instalirati valjani JDK (Java Development Kit) i preuzeti i pokrenuti NiFi.

### 11.1. Java Development Kit(JDK)

#### 11.1.1. Preuzimanje JDK-a

U ovom dijelu pretpostavit ćemo da nemate instaliranu JDK verziju. Na Windows računalu možete jednostavno provjeriti: idite na Start > Upravljačka ploča > Programi i provjerite je li Java instalirana. Također, možete otvoriti naredbeni redak i unijeti "javac". Ako se naredba ne prepoznaje, vrlo je vjerojatno da nemate JDK (mada to može ovisiti o postavkama varijabli sustava, što ćemo kasnije vidjeti).

Za preuzimanje JDK-a, posjetite web stranicu tvrtke Oracle. U mojem primjeru, odabrao sam instalirati JDK 11.

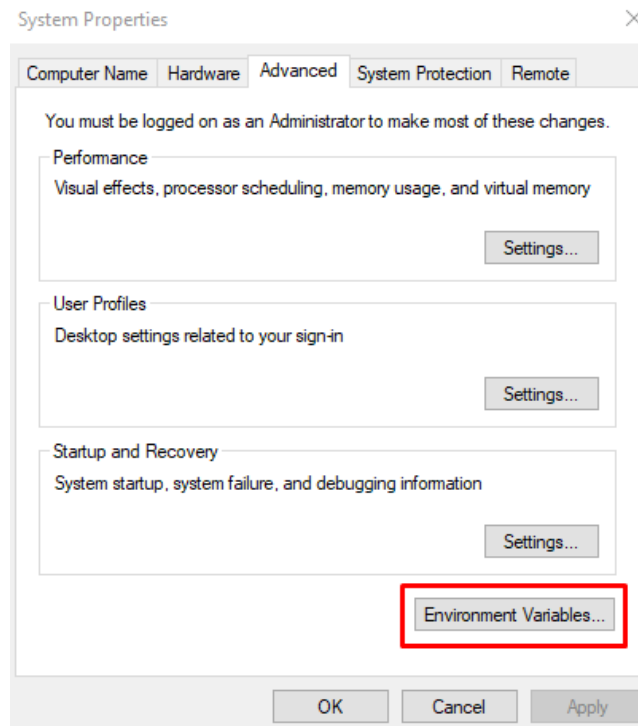
| ORACLE                           |           |  | Products Industries Resources Customers Partners Developers Company | Q | View Accounts | Contact Sales |
|----------------------------------|-----------|--|---|---|---------------|---------------|
| Linux ARM 64 Compressed Archive  | 157.62 MB | <a href="#">jdk-11.0.18_linux-aarch64_bin.tar.gz</a>   |   |   |               |               |
| Linux x64 Debian Package         | 138.86 MB | <a href="#">jdk-11.0.18_linux-x64_bin.deb</a>          |   |   |               |               |
| Linux x64 RPM Package            | 161.37 MB | <a href="#">jdk-11.0.18_linux-x64_bin.rpm</a>          |   |   |               |               |
| Linux x64 Compressed Archive     | 161.48 MB | <a href="#">jdk-11.0.18_linux-x64_bin.tar.gz</a>       |   |   |               |               |
| macOS Arm 64 Compressed Archive  | 153.75 MB | <a href="#">jdk-11.0.18_macos-aarch64_bin.tar.gz</a>   |   |   |               |               |
| macOS Arm 64 DMG Installer       | 153.24 MB | <a href="#">jdk-11.0.18_macos-aarch64_bin.dmg</a>      |   |   |               |               |
| macOS x64 Compressed Archive     | 155.89 MB | <a href="#">jdk-11.0.18_macos-x64_bin.tar.gz</a>       |   |   |               |               |
| macOS x64 DMG Installer          | 155.38 MB | <a href="#">jdk-11.0.18_macos-x64_bin.dmg</a>          |   |   |               |               |
| Solaris SPARC Compressed Archive | 185.03 MB | <a href="#">jdk-11.0.18_solaris-sparcv9_bin.tar.gz</a> |   |   |               |               |
| <b>Windows x64 Installer</b>     | 141.14 MB | <a href="#">jdk-11.0.18_windows-x64_bin.exe</a>        |   |   |               |               |
| Windows x64 Compressed Archive   | 158.85 MB | <a href="#">jdk-11.0.18_windows-x64_bin.zip</a>        |   |   |               |               |

Slika 46: JDK download stranica(Izvor: vlastita izrada)

Kada preuzmete datoteku, dvaput kliknite na nju i pratite upute čarobnjaka. Neće biti potrebe za mijenjanjem bilo čega, stoga samo potvrdite i kliknite na "Dalje" kada god se od vas zatraži, te pričekajte da se instalacija dovrši. Zapamtite naziv mape u koju će JDK biti instaliran (obično nešto poput "C:\Program Files\Java").

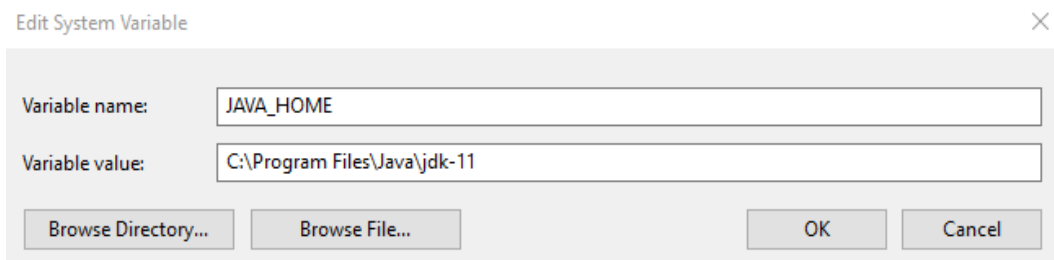
### 11.1.2. Postavljanje varijabli sustava

Kada završi instalacija, trebat će vam uređivanje varijabli sustava kako bi vaše prijenosno računalo moglo pronaći JDK lokaciju kada je to potrebno. Idite na Start izbornik, potražite "Uredi varijable okruženja sustava" (Edit the system environment) i otvorite karticu "Svojstva sustava". Zatim, na dnu, kliknite na "Varijable okruženja".



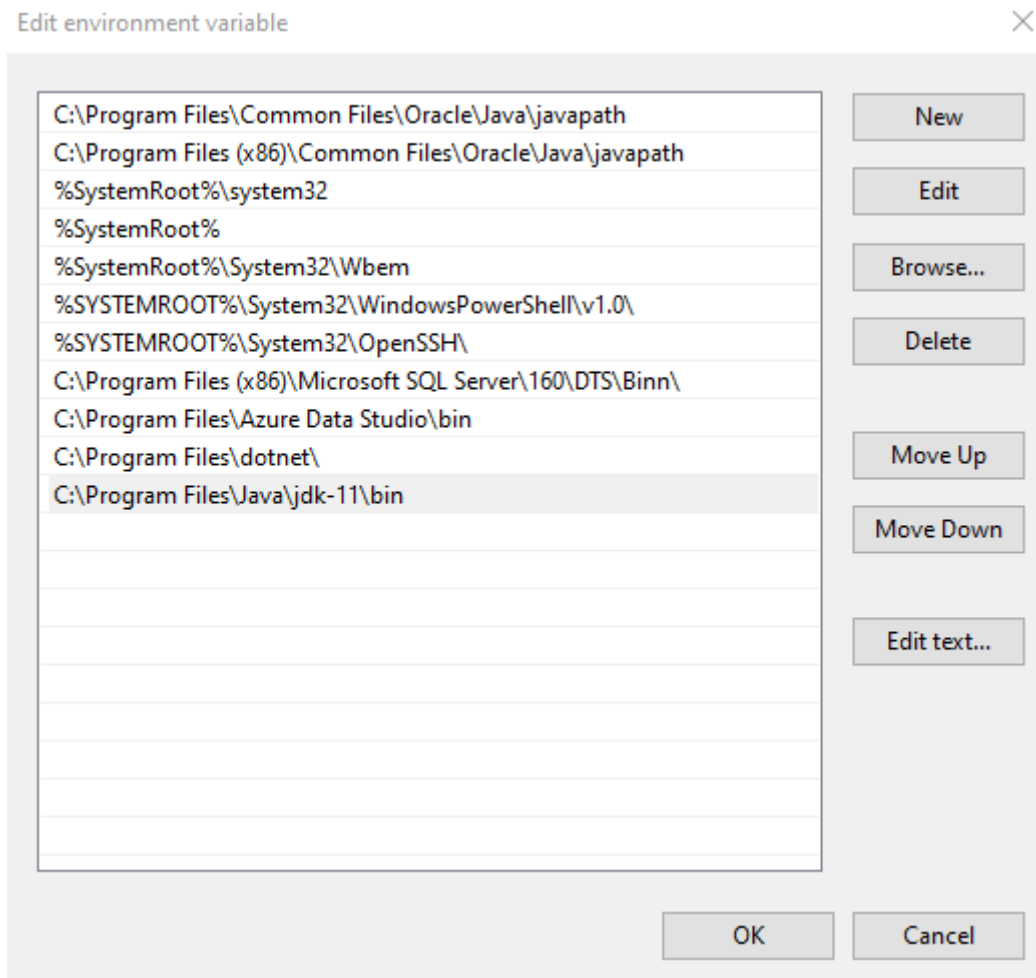
Slika 47: Dodavanje varijable okruženja(Izvor: vlastita izrada)

Kliknite Novo na dnu kartice koja se otvori za dodavanje nove sistemske varijable. Naš smo nazvali JAVA\_HOME, s vrijednošću gdje je instaliran JDK (u našem slučaju, C:\Program Files\Java\jdk-11).



Slika 48: Dodavanje sistemske varijable(Izvor: vlastita izrada)

Zatim se pomaknite niz popis postojećih varijabli dok ne pronađete 'Path'; otvorite ga i na dnu dodajte novi redak, koristeći isti put kao prije, ali ovaj put sa sufiksom “bin” (dakle, u našem slučaju, “C:\Program Files\Java\jdk-11 \bin”).



Slika 49: Prikaz nakon dodavanja varijable(Izvor: vlastita izrada)

### 11.1.3. Provjera instalacije JDK-a

U ovom trenutku, JDK bi trebao biti instaliran i prepoznat od strane vašeg računala; kako biste to potvrdili, otvorite naredbeni redak i unesite "javac". Ako izlaz izgleda kao na slici ispod, tada ste spremni!

```
Microsoft Windows [Version 10.0.19045.3086]
(c) Microsoft Corporation. All rights reserved.

C:\Users\igorn>javac
Usage: javac <options> <source files>
where possible options include:
-g Generate all debugging info
-g:none Generate no debugging info
-g:{lines,vars,source} Generate only some debugging info
-nowarn Generate no warnings
-verbose Output messages about what the compiler is doing
-deprecation Output source locations where deprecated APIs are used
-classpath <path> Specify where to find user class files and annotation processors
-cp <path> Specify where to find user class files and annotation processors
-sourcepath <path> Specify where to find input source files
-bootclasspath <path> Override location of bootstrap class files
-extdirs <dirs> Override location of installed extensions
-endorseddirs <dirs> Override location of endorsed standards path
-processor:{none,only} Control whether annotation processing and/or compilation is done.
-processor <class1>[,<class2>,<class3>...] Names of the annotation processors to run; bypasses default discovery process
-processorpath <path> Specify where to find annotation processors
-parameters Generate metadata for reflection on method parameters
-d <directory> Specify where to place generated class files
-s <directory> Specify where to place generated source files
-h <directory> Specify where to place generated native header files
-implicit:{none,class} Specify whether or not to generate class files for implicitly referenced files
-encoding <encoding> Specify character encoding used by source files
-source <release> Provide source compatibility with specified release
-target <release> Generate class files for specific VM version
-profile <profile> Check that API used is available in the specified profile
-version Version information
-help Print a synopsis of standard options
-Akey[=value] Options to pass to annotation processors
-X Print a synopsis of nonstandard options
-><flag> Pass <flag> directly to the runtime system
-Werror Terminate compilation if warnings occur
@<filename> Read options and filenames from file
```

Slika 50: Provjera instalacije JDK-a(Izvor: vlastita izrada)

## 11.2. Apache Nifi

### 11.2.1. Preuzimanje Apache Nifi-a

Sada kada imamo funkcionalni JDK, možemo prijeći na preuzimanje i instaliranje NiFi-a. Da biste preuzeli NiFi, idite na ovaj [link](#) (ili jednostavno guglajte "download nifi").



Slika 51: Stranica za preuzimanje Nifi-a (Izvor: vlastita izrada)

Verzija koju želimo završava s "bin.zip" (zip mapa binarne datoteke). Kliknite na nju i bit ćete prebačeni na drugu stranicu gdje zapravo možete preuzeti datoteku (koristeći prvu vezu).

### 11.2.2. Instalacija

Nakon što se zip datoteka preuzme, samo je raspakirajte. Jednostavan i učinkovit izbor je raspakirati mapu u C:\, i trebali biste završiti sa stazom poput C:\nifi-1.16.3, u kojoj biste već trebali vidjeti bin direktorij.

Sada morate otići do te bin mape i dvaput kliknuti datoteku run-nifi.bat.

| Name        | Date modified    | Type               | Size  |
|-------------|------------------|--------------------|-------|
| dump-nifi   | 22-01-2020 15:10 | Windows Batch File | 2 KB  |
| nifi.sh     | 22-01-2020 15:10 | SH File            | 13 KB |
| nifi-env    | 22-01-2020 15:10 | Windows Batch File | 2 KB  |
| nifi-env.sh | 22-01-2020 15:10 | SH File            | 2 KB  |
| run-nifi    | 22-01-2020 15:10 | Windows Batch File | 2 KB  |
| status-nifi | 22-01-2020 15:10 | Windows Batch File | 2 KB  |

Slika 52: Pokretanje Nifi-a(Izvor: vlastita izrada)

Run-nifi.bat će pokrenuti naredbeni redak, koji će izgledati kao na slici ispod.

```

C:\WINDOWS\system32\cmd.exe
~1.4\.\lib\javax.servlet-api-3.1.0.jar;C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\.\lib\jcl-over-slf4j-1.7.30.jar;C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\.\lib\jetty-schemas-3.1.jar;C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\.\lib\jul-to-slf4j-1.7.30.jar;C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\.\lib\log4j-over-slf4j-1.7.30.jar;C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\.\lib\logback-classic-1.2.3.jar;C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\.\lib\logback-core-1.2.3.jar;C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\.\lib\nifi-api-1.11.4.jar;C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\.\lib\nifi-framework-api-1.11.4.jar;C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\.\lib\nifi-nar-utils-1.11.4.jar;C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\.\lib\nifi-properties-1.11.4.jar;C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\.\lib\nifi-runtime-1.11.4.jar;C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\.\lib\slf4j-api-1.7.30.jar -Dorg.apache.jasper.compile.disablejsr199=true -Xmx512m -Xms512m -Djavax.security.auth.useSubjectCredsOnly=true -Djava.security.egd=file:/dev/urandom -Dsun.net.http.allowRestrictedHeaders=true -Djava.net.preferIPv4Stack=true -Djava.awt.headless=true -Djava.protocol.handler.pkgs=sun.net.www.protocol -Dzookeeper.admin.enableServer=false -Dnifi.properties.file.path=C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\.\conf\nifi.properties -Dnifi.bootstrap.listen.port=50604 -Dapp=NiFi -Dorg.apache.nifi.bootstrap.config.log.dir=C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\bin\..\logs org.apache.nifi.NiFi
2020-04-28 15:01:39,913 WARN [main] org.apache.nifi.bootstrap.Command Failed to set permissions so that only the owner can read pid file C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\bin\.\run\nifi.pid; this may allow others to have access to the key needed to communicate with NiFi. Permissions should be changed so that only the owner can read this file
2020-04-28 15:01:40,344 WARN [main] org.apache.nifi.bootstrap.Command Failed to set permissions so that only the owner can read status file C:\Users\VISHAL~1\DOWNLO~1\NIFI-1~1.4\bin\.\run\nifi.status; this may allow others to have access to the key needed to communicate with NiFi. Permissions should be changed so that only the owner can read this file
2020-04-28 15:01:40,990 INFO [main] org.apache.nifi.bootstrap.Command Launched Apache NiFi with Process ID 7712

```

Slika 53: Prikaz nakon pokretanja Nifi-a(Izvor: vlastita izrada)

### 11.2.3. Provjera porta

NiFi je pokrenut. Sada ga možete otvoriti iz bilo kojeg web preglednika kao što je Chrome ili Internet Explorer itd. Dakle, potreban nam je broj porta za pokretanje NiFi korisničkog sučelja na web pregledniku.

Stoga idite u mapu conf (nifi-1.22.0/conf) koja sadrži sve konfiguracijske datoteke za



NiFi i otvorite datoteku nifi.properties u notepadu. Pomaknite se prema dolje i provjerite broj priključka nifi.

```
nifi.web.https.host=127.0.0.1
nifi.web.https.port=8443
nifi.web.https.network.interface.default=
nifi.web.https.application.protocols=http/1.1
nifi.web.jetty.working.directory=./work/jetty
nifi.web.jetty.threads=200
nifi.web.max.header.size=16 KB
nifi.web.proxy.context.path=
nifi.web.proxy.host=
nifi.web.max.content.size=
nifi.web.max.requests.per.second=30000
nifi.web.max.access.token.requests.per.second=25
nifi.web.request.timeout=60 secs
nifi.web.request.ip.whitelist=|
nifi.web.should.send.server.version=true
nifi.web.request.log.format=%{client}a - %u %t "%r" %s %0 "%{Referer}i" "%{User-Agent}i"
```

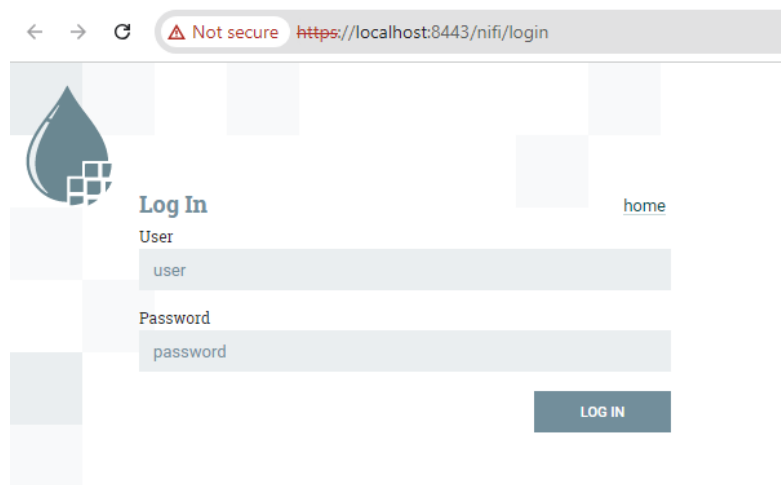
Slika 54:Provjera porta(Izvor: vlastita izrada)

Zadani port Apache NiFi je 8443. Ako je zadani broj porta već dodijeljen nekom drugom softverskom programu, promijenite broj porta kao što je 9090 i spremite datoteku.

Zadana instalacija generira nasumično korisničko ime i lozinku, zapisujući generirane vrijednosti u dnevnik aplikacije. Dnevnik aplikacije nalazi se u logs/nifi-app.log u instalacijskom direktoriju. Dnevnik će sadržavati retke s generiranim korisničkim imenom [USERNAME] i generiranom lozinkom [PASSWORD] koji označavaju vjerodajnice potrebne za pristup. Potražite te retke u zapisniku aplikacije i zabilježite generirane vrijednosti na sigurno mjesto.

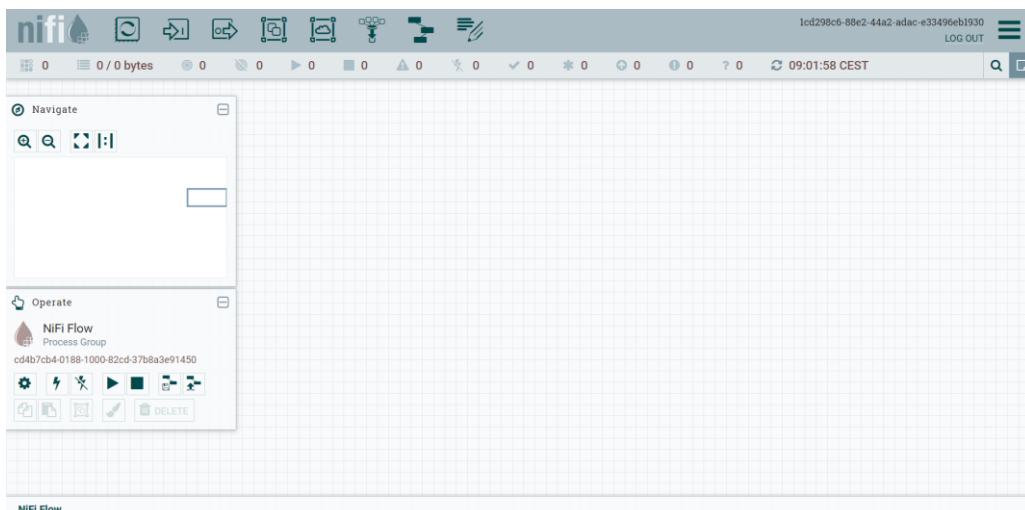
Također, web preglednik će prikazati poruku upozorenja koja ukazuje na potencijalni sigurnosni rizik zbog samopotpisanog certifikata NiFi generiranog tijekom inicijalizacije. Prihvatanje potencijalnog sigurnosnog rizika i nastavak učitavanja sučelja opcija je za razvojne instalacije. Samopotpisani certifikat ističe nakon 60 dana. Proizvodne implementacije trebale bi osigurati certifikat od pouzdanog tijela i ažurirati NiFi spremište ključeva i konfiguraciju pouzdanog spremišta.

## 11.2.4. Pokretanje Apache Nifi-a



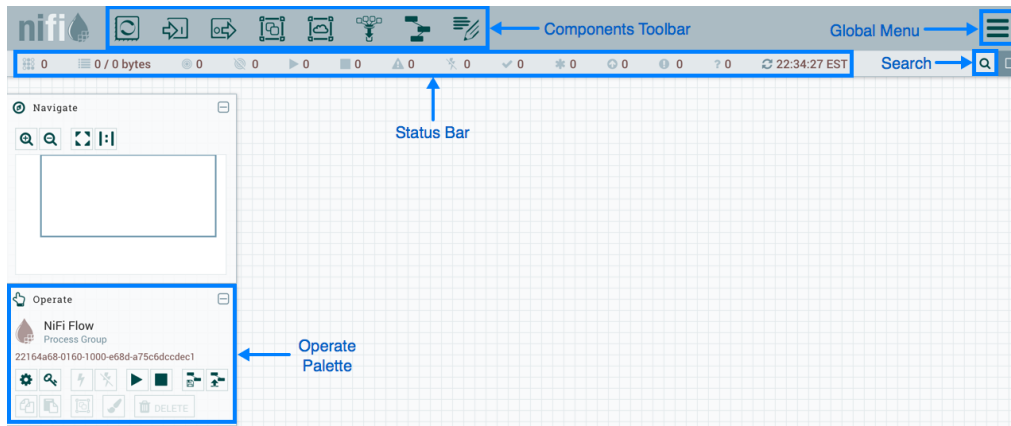
Slika 55: Početni zaslon Nifi-a(Izvor: vlastita izrada)

Koristeći generirane vjerodajnice, unesite generirano korisničko ime u polje Korisnik i generiranu lozinku u polje Lozinka, zatim odaberite PRIJAVA za pristup sustavu. Ovo će dovesti do korisničkog sučelja, koje je u ovom trenutku prazno platno za orkestriranje protoka podataka:



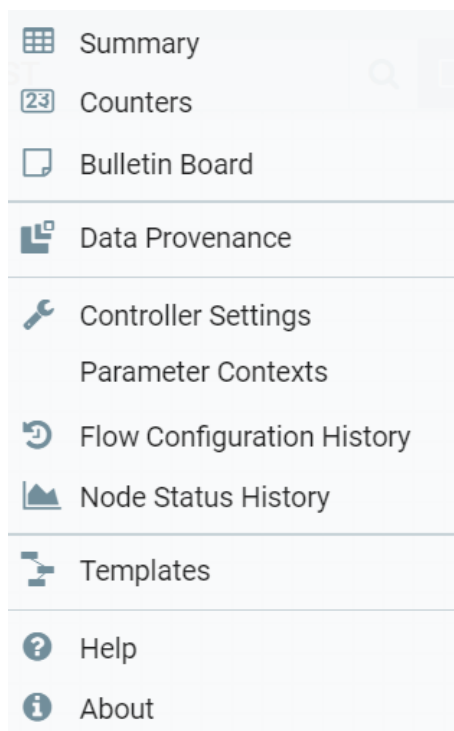
Slika 56: Sučelje Nifi-a(Izvor: vlastita izrada)

UI ima više alata za stvaranje i upravljanje prvim protokom podataka:



Slika 57: Prikaz alatnih traka i postavka(Izvor: vlastita izrada)

Također, imamo globalni izbornik koji sadrži sljedeće opcije:



Slika 58:Globalni izbornik(Izvor: vlastita izrada)