**Selection of the optimal drug for a patient with a complex set of diagnoses**

**Authors:** Olga Katkov

**Date:** 2022/20/09



**Contents**

[**Introduction & Business Idea** 3](#_Toc115260024)

[**Data sources & Toolbox** 5](#_Toc115260025)

[**Goals** 6](#_Toc115260026)

[**Solution & doctor Assistence application main functions** 7](#_Toc115260027)

[1.Doctor Application Functionality 7](#_Toc115260028)

[2.Application Logic Flow: 8](#_Toc115260029)

[3.NIFI Process steps & Flow 8](#_Toc115260030)

[4.Kafka 11](#_Toc115260031)

[5. Application Classes 12](#_Toc115260032)

[**Data Manipulation & Data Structure** 12](#_Toc115260033)

[API DATA. 12](#_Toc115260034)

[2.Doctor db ERD 16](#_Toc115260035)

[3.Receipt’s\_Archive 16](#_Toc115260036)

[4. Hive Table Structure, Hive db 16](#_Toc115260037)

[5. Mango db 17](#_Toc115260038)

[6. Lexical analysis and data transformation 17](#_Toc115260039)

[**Architecture implemented getDrug Interaction** 20](#_Toc115260040)

[**Possible reports and queries** 21](#_Toc115260041)

[**Scripts examples** 21](#_Toc115260042)

# **Introduction & Business Idea**

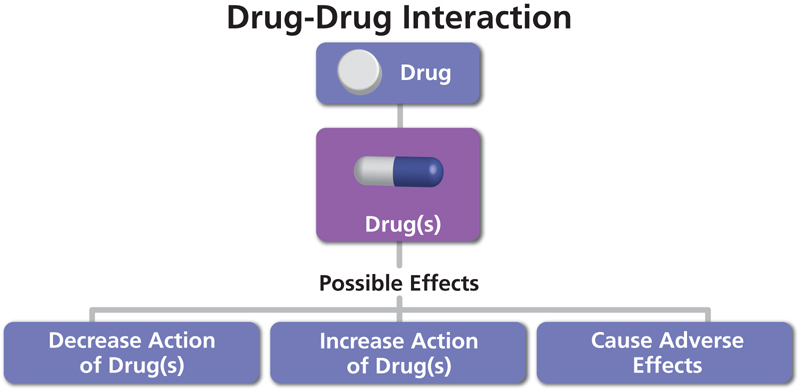
**Medical errors officially the third leading cause of death in U.S., study finds.**

a large proportion of these deaths are due to improper selection of drugs for chronic patients and with a wide range of constantly taken drugs.

Possible full new drug checks include 4 types of interactions:



* In this project only one type Drug-Drug interaction will be checked. In this project developed automatic checking of a new drug recommended by a doctor for a patient for interaction with constantly taken drugs. Interaction can be both positive (synergy) and negative, harming the patient. To warn the doctor about the possible negative effects of the new drug is the goal of this project.
* Drug Antagonism
* Drug Interaction
* Drug Synergism



For this purpose will be used NIH supplied some drug interaction APIs…

https://lhncbc.nlm.nih.gov/RxNav/APIs/InteractionAPIs.html

<https://rxnav.nlm.nih.gov/REST/rxcui.json?name=Aspirin>

https://rxnav.nlm.nih.gov/REST/interaction/interaction.json?rxcui=88014&sources=DrugBank

# **Data sources & Toolbox**

|  |  |  |
| --- | --- | --- |
| **Source** | **Typ &e** | **Description** |
| Doctor DB | MySql , Relation DB | Rt doctor db ,that saved patient history, doctors information. |
| Diseases DB | constantly taken drugs | Loaded data from external data source |
| External APIs –drugs api | 4 official drug interaction apis of National Institute of Health, USA | https://lhncbc.nlm.nih.gov/RxNav/APIs/InteractionAPIs.html |
| All drug receipts archive | HDFS, cloud |  |
| Mongo db | New db to save data in useful format after first checks for further uses without |  |

**Toolbox for technical realization**

|  |  |  |
| --- | --- | --- |
| **Tool** | **use** | **description** |
| MySql | 2 databases | 1.Loaded from external source diseases database  2. Doctor database(Kupat holim db) with his patient’s  history , doctor data, recepts etc |
| kafka | Data stream tool , work with 2 types of topics  topic1 = 'GetDrug' topic2 = 'GetInteraction' | 1.Topic –trigger of start process by run  kafka producer, consumer in nifi  after receiving request start the check process  2.Topic from second type in doctor application run  consumer to check result , the producer  will be the last step on nifi process |
| nifi | mechanism of work. Receiving and processing data. The result is sent via kafka to the doctor's application | Consumer of topic 1 receive the request. Run 1st api call  to get drugId according to name.  2. Run 2d api call to get interaction according Id  3. Jolt data manipulation to simplify the data from api  4. Send result to kafka with topic 2 type |
| spark | mechanism of work. Receiving and processing data. The result is sent via kafka to the doctor's application |  |
| **hdfs** | **Receipts data archive** | **Source for reports and analytics** |
| NIH apis | Source DrugBank | Api can works with different data source,  in this project we use the DrugBank source |
| python |  | Use parrow, pyspark , mysql, kafka, pymongo |
| HIVE | Sql tools for reports from hdfs archive | Used to get reports from archive |
| mongo | New database to save the result of data manipulation | For future use, after first period of application use  possible to use ready data sets for interaction instead of  run the whole process each time. Full process will be  needed in future only when  the drug don’t exist in mongo db. |

# **Goals**

Our purpose is

1.to check all health chronic history of patient, all drugs that patient get on constant way to conflict or synergy to new recommended by doctor drug and

2.to give to doctor as result all constant drugs from patient treatment list in interaction of new recommended by doctor drug with warning on all possible negative interaction and its severity

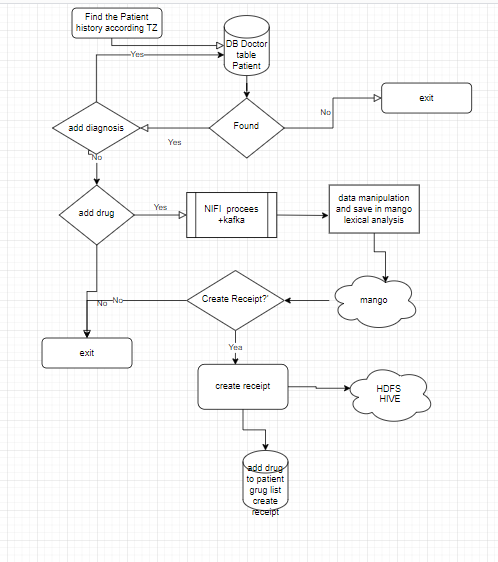
3. if the doctor want to give the receipt add all decision history to central archive of the country that allows analytics by ministry of health about all receipt that can cause to unwanted side effects and to increase risks to chronic patients.

# **Solution & doctor Assistence application main functions**

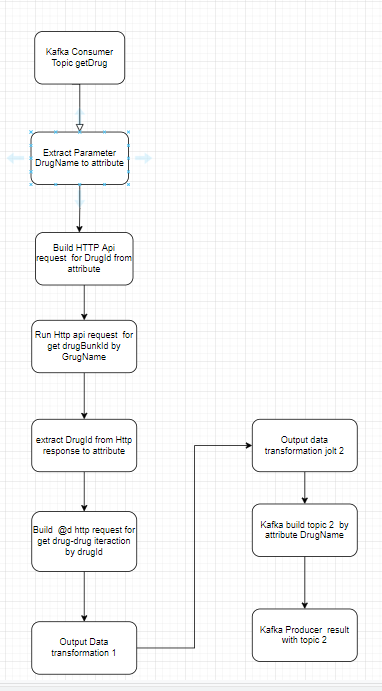
## 1.Doctor Application Functionality

|  |  |  |
| --- | --- | --- |
| Visit | Function method | description |
| btnCrRicept | CreateReceipt | 1.Add drug to constant drug list and update Patient in mySql  2. save all data in Hdfs with list of Patient diseases and list of constant drug  3. Load in path to hive dir |
| btnPatient | findPatient | Find the patient according tz in mySql db, get Patient data  2. fill diseases df and and constant drugs df |
| btnDiag | addDiagnose | Add diagnosis to existing list and update in Mysql |
| btnDrug | checkDrug | 1. Run consumer GetInteraction$DrugNume  2. Kafka producer on GetDrug topic  3. once Receive consumer data  3.1 Stop consumer  3.2. remove from drug list id and drug name list source drug values(exclude from list)  3.3. Build synergy or contraindications and severity according to lexica analysis and  3.4 write result to mango  3.4. Build result df and show it |
| Error log |  | Mysql error table |

## 2.Application Logic Flow:



## 3.NIFI Process steps & Flow

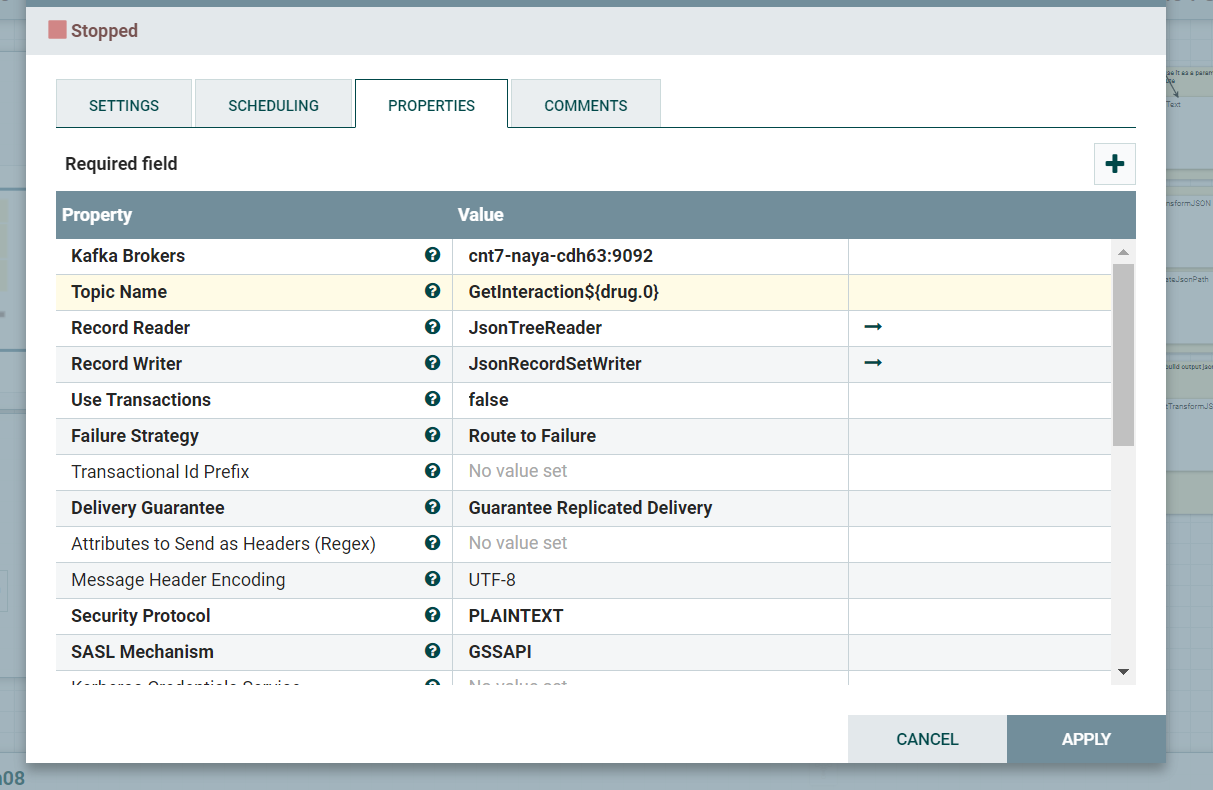


|  |  |  |
| --- | --- | --- |
| Proccessor Type | Parameters | Description |
| **ConsumeKafka\_2\_6 1.14.0** | Topic=GetDrug | receive getDrug requests from clients with specific drug name as parameter  topic=GetDrug |
| **ExtractText 1.14.0** |  | extract the drug name and use it as a parameter on api call,  saved as nifi process attribute |
| **InvokeHTTP 1.14.0** | https://rxnav.nlm.nih.gov/REST/rxcui.json?name=${drug.0}&search=0 | Get Drugid from API |
| **JoltTransformJSON 1.14.0** | [{  "operation": "shift",  "spec": {  "idGroup": {  "rxnormId":"rxnormId"  }}}] | get only value drug info from http response |
| **JoltTransformJSON 1.14.0** | [{"operation": "modify-overwrite-beta","spec": {    "rxnormId": "=lastElement(@(1,rxnormId))"    }}] | cut id only from json |
| **EvaluateJsonPath 1.14.0** | drug\_id=$.rxnormId | put drug\_id to attribute drug\_id |
| **InvokeHTTP 1.14.0** | https://rxnav.nlm.nih.gov/REST/interaction/interaction.json?rxcui=${drug\_id}&sources=DrugBank | http api call to get drug interaction according id in attribute |
| **JoltTransformJSON 1.14.0** | [{  "operation": "shift",  "spec": {  "interactionTypeGroup": {  "\*": {  "interactionType": {  "\*": {  "minConceptItem": {  "rxcui": "rxcui",  "name": "name"  },  "interactionPair": {  "\*": {  "severity": "interactionPair.severity",  "description": "interactionPair.description",  "interactionConcept": {  "\*": {  "minConceptItem": {  "rxcui": "interactionPair.ConceptItem.rxcui",  "name": "interactionPair.ConceptItem.name"  } } } }  }  }  }  }  }  }  }] | Get only relevant data from http response |
| **PublishKafkaRecord\_2\_6 1.14.0** | Topic= GetInteraction${drug.0} | Send grug interaction result to doctor application |

## 4.Kafka

Our project contains 3 topics:

* topic1 = GetDrug’
* topic2 = “GetInteraction” +DrugName



## 5. Application Classes

|  |  |
| --- | --- |
| **Files Or Classes** | **methods** |
| ClassReciept | crReceipt |
| PatientClass | 1. addDiagnose  2. addDrugToPatientDrugList  3. findPatient |
| ProducerReq | send |
| configuration |  |
| Ui\_methods | checkDrugKf  ClassifyDesc |

# **Data Manipulation & Data Structure**

## API DATA.

1. Full api result date- large and not useful format . **Large and not useful file format. The example saved in github**

[**https://github.com/katkovOlga/nayaProject/blob/main/Dr\_dr\_Out.txt**](https://github.com/katkovOlga/nayaProject/blob/main/Dr_dr_Out.txt)

**It seems so, followed part of api result:**

{"nlmDisclaimer":"It is not the intention of NLM to provide specific medical advice, but rather to provide users with information to better understand their health and their medications. NLM urges you to consult with a qualified physician for advice about medications.","interactionTypeGroup":[{"sourceDisclaimer":"DrugBank is intended for educational and scientific research purposes only and you expressly acknowledge and agree that use of DrugBank is at your sole risk. The accuracy of DrugBank information is not guaranteed and reliance on DrugBank shall be at your sole risk. DrugBank is not intended as a substitute for professional medical advice, diagnosis or treatment..[www.drugbank.ca]","sourceName":"DrugBank","interactionType":[{"comment":"aspirin (1191) is resolved to aspirin (1191)","minConceptItem":{"rxcui":"1191","name":"aspirin","tty":"IN"},"interactionPair":[{"interactionConcept":[{"minConceptItem":{"rxcui":"1191","name":"aspirin","tty":"IN"},"sourceConceptItem":{"id":"DB00945","name":"Acetylsalicylic acid","url":"https://go.drugbank.com/drugs/DB00945#interactions"}},{"minConceptItem":{"rxcui":"1000492","name":"resveratrol","tty":"IN"},"sourceConceptItem":{"id":"DB02709","name":"Resveratrol","url":"https://go.drugbank.com/drugs/DB02709#interactions"}}],"severity":"N/A","description":"Acetylsalicylic acid may increase the antiplatelet activities of Resveratrol."},{"interactionConcept":[{"minConceptItem":{"rxcui":"1191","name":"aspirin","tty":"IN"},"sourceConceptItem":{"id":"DB00945","name":"Acetylsalicylic acid","url":"https://go.drugbank.com/drugs/DB00945#interactions"}},{"minConceptItem":{"rxcui":"1001","name":"antipyrine","tty":"IN"},"sourceConceptItem":{"id":"DB01435","name":"Antipyrine","url":"https://go.drugbank.com/drugs/DB01435#interactions"}}],"severity":"N/A","description":"The therapeutic efficacy of Acetylsalicylic acid can be decreased when used in combination with Antipyrine."},{"interactionConcept":[{"minConceptItem":{"rxcui":"1191","name":"aspirin","tty":"IN"},"sourceConceptItem":{"id":"DB00945","name":"Acetylsalicylic acid","url":"https://go.drugbank.com/drugs/DB00945#interactions"}},{"minConceptItem":{"rxcui":"1005911","name":"influenza virus vaccine, live attenuated, A-Perth-16-2009 (H3N2) strain","tty":"IN"},"sourceConceptItem":{"id":"DB14449","name":"Influenza A virus A/Perth/16/2009 (H3N2) live (attenuated) antigen","url":"https://go.drugbank.com/drugs/DB14449#interactions"}}],"severity":"N/A","description":"The risk or severity of adverse effects can be increased when Influenza A virus A/Perth/16/2009 (H3N2) live (attenuated) antigen is combined with Acetylsalicylic acid."},{"interactionConcept":[{"minConceptItem":{"rxcui":"1191","name":"aspirin","tty":"IN"},"sourceConceptItem":{"id":"DB00945","name":"Acetylsalicylic acid","url":"https://go.drugbank.com/drugs/DB00945#interactions"}},{"minConceptItem":{"rxcui":"1006469","name":"doconexent","tty":"IN"},"sourceConceptItem":{"id":"DB03756","name":"Doconexent","url":"https://go.drugbank.com/drugs/DB03756#interactions"}}],"severity":"N/A","description":"The metabolism of Acetylsalicylic acid can be decreased when combined with Doconexent."},{"interactionConcept":[{"minConceptItem":{"rxcui":"1191","name":"aspirin","tty":"IN"},"sourceConceptItem":{"id":"DB00945","name":"Acetylsalicylic acid","url":"https://go.drugbank.com/drugs/DB00945#interactions"}},{"minConceptItem":{"rxcui":"1009","name":"antithrombin III","tty":"IN"},"sourceConceptItem":{"id":"DB11598","name":"Antithrombin III human","url":"https://go.drugbank.com/drugs/DB11598#interactions"}}],"severity":"N/A","description":"Acetylsalicylic acid may increase the anticoagulant activities of Antithrombin III human."},{"interactionConcept":[{"minConceptItem":{"rxcui":"1191","name":"aspirin","tty":"IN"},"sourceConceptItem":{"id":"DB00945","name":"Acetylsalicylic acid","url":"https://go.drugbank.com/drugs/DB00945#interactions"}},{"minConceptItem":{"rxcui":"10106","name":"streptokinase","tty":"IN"},"sourceConceptItem":{"id":"DB00086","name":"Streptokinase","url":"https://go.drugbank.com/drugs/DB00086#interactions"}}],"severity":"N/A","description":"Acetylsalicylic acid may increase the anticoagulant activities of Streptokinase."},{"interactionConcept":[{"minConceptItem":{"rxcui":"1191","name":"aspirin","tty":"IN"},"sourceConceptItem":{"id":"DB00945","name":"Acetylsalicylic acid","url":"https://go.drugbank.com/drugs/DB00945#interactions"}},{"minConceptItem":{"rxcui":"10109","name":"streptomycin","tty":"IN"},"sourceConceptItem":{"id":"DB01082","name":"Streptomycin","url":"https://go.drugbank.com/drugs/DB01082#interactions"}}],"severity":"N/A","description":"Acetylsalicylic acid may decrease the excretion rate of Streptomycin which could result in a higher serum level."},{"interactionConcept":[{"minConceptItem":{"rxcui":"1191","name":"aspirin","tty":"IN"},"sourceConceptItem":{"id":"DB00945","name":"Acetylsalicylic acid","url":"https://go.drugbank.com/drugs/DB00945#interactions"}},{"minConceptItem":{"rxcui":"10114","name":"streptozocin","tty":"IN"},"sourceConceptItem":{"id":"DB00428","name":"Streptozocin","url":"https://go.drugbank.com/drugs/DB00428#interactions"}}],"severity":"N/A","description":"The risk or severity of bleeding can be increased when Acetylsalicylic acid is combined with Streptozocin."},{"interactionConcept":[{"minConceptItem":{"rxcui":"1191","name":"aspirin","tty":"IN"},"sourceConceptItem":{"id":"DB00945","name":"Acetylsalicylic acid","url":"https://go.drugbank.com/drugs/DB00945#interactions"}},{"minConceptItem":{"rxcui":"10154","name":"succinylcholine","tty":"IN"},"sourceConceptItem":{"id":"DB00202","name":"Succinylcholine","url":"https://go.drugbank.com/drugs/DB00202#interactions"}}],"severity":"N/A","description":"The risk or severity of hyperkalemia can be increased when Succinylcholine is combined with Acetylsalicylic acid."},{"interactionConcept":[{"minConceptItem":{"rxcui":"1191","name":"aspirin","tty":"IN"},"sourceConceptItem":{"id":"DB00945","name":"Acetylsalicylic acid","url":"https://go.drugbank.com/drugs/DB00945#interactions"}},{"minConceptItem":{"rxcui":"10167","name":"sulbactam","tty":"IN"},"sourceConceptItem":{"id":"DB09324","name":"Sulbactam","url":"https://go.drugbank.com/drugs/DB09324#interactions"}}],"severity":"N/A","description":"Acetylsalicylic acid may decrease the excretion rate of Sulbactam which could result in a higher serum level."}

….

Need be splited to only need results. Path in large output to required results is:

interactionTypeGroup.interactionType.minConceptItem.rxcui

interactionTypeGroup.interactionType.minConceptItem.name

[

interactionTypeGroup.interactionType.interactionPair.interactionConcept[1].minConceptItem.rxcui

interactionTypeGroup.interactionType.interactionPair.interactionConcept[1].minConceptItem.name

interactionTypeGroup.interactionType.interactionPair.severity

interactionTypeGroup.interactionType.interactionPair.description

Jolt Specification:

[{

"operation": "shift",

"spec": {

"interactionTypeGroup": {

"\*": {

"interactionType": {

"\*": {

"minConceptItem": {

"rxcui": "rxcui",

"name": "name"

},

"interactionPair": {

"\*": {

"severity": "severity",

"description": "description",

"interactionConcept": {

"\*": {

"minConceptItem": {

"rxcui": "IdList",

"name": "NameList"

} } } } } } } } } }}]

Final Nifi produced result:

{

"rxcui" : "88014",

"name" : "rizatriptan",

"severity" : [ "high", "high", "high", "high", "high" ],

"description" : [ "Triptans - monoamine oxidase (MAO) inhibitors", "Triptans - monoamine oxidase (MAO) inhibitors", "Triptans - monoamine oxidase (MAO) inhibitors", "Triptans - monoamine oxidase (MAO) inhibitors", "Triptans - monoamine oxidase (MAO) inhibitors" ],

"IdList" : [ "88014", "10734", "88014", "30121", "88014", "6011", "88014", "6878", "88014", "8123" ],

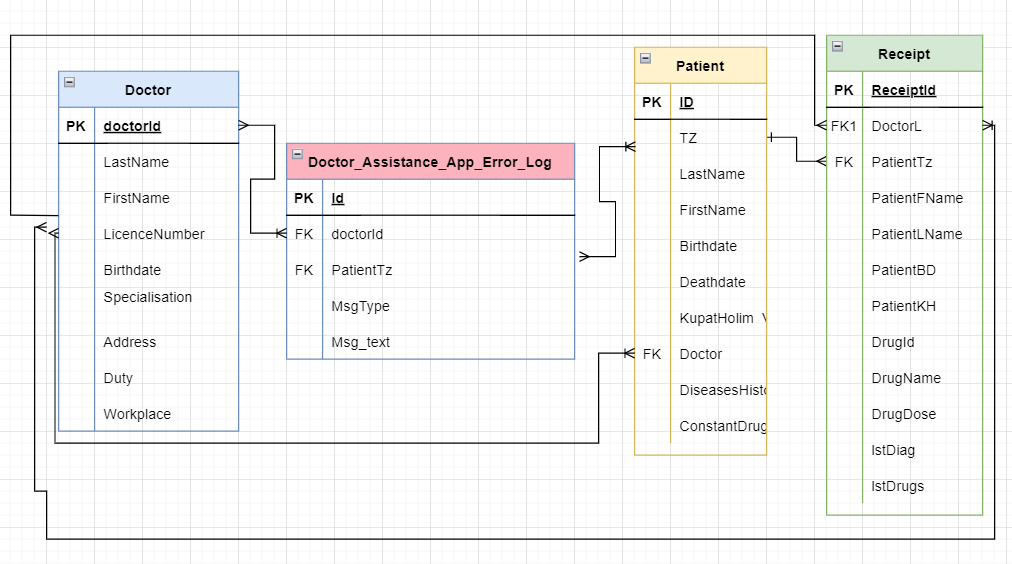
"NameList" : [ "rizatriptan", "tranylcypromine", "rizatriptan", "moclobemide", "rizatriptan", "isocarboxazid", "rizatriptan", "methylene blue", "rizatriptan", "phenelzine" ]

}

Result JSON Structure

|  |  |  |
| --- | --- | --- |
| Field | Description | use |
| rxcui | DrugBankId | As parameter to interaction api call |
| name | Drug name | Parameter for first api call to get drug bank id |
| severity | List of known interaction severity | Not exist in free api |
| description | List of interaction descriptions | Used by lexical analysis to understand severity |
| NameList | Names of drugs with known interaction | Used to compare with patient constant treatment drug list |
| IdList | Drug Banks ids of known interactions |  |

## 2.Doctor db ERD



## 3.Receipt’s\_Archive

* . The data will be saved as json file.The data will be saved in Parquet format
* df =pd.DataFrame(data={"DoctorLicense" : [self.DoctorL], "PatientTZ": [self.PatientTz], "PatientFirstName": [self.PatientFName]\  
   , "ConflictSExists": ConflictSExists,"PatientLastName": [self.PatientLName], "KupatHolim": [self.PatientKH], "PtienBirthdate": [self.PatientBD] \  
   , "DrugName": [self.DrugName] , "DrugId": [self.DrugId] ,"DrugDose": [self.DrugDose] , "dateCreated": [currDT] \  
   ,"PatientDiseasesLqist": [self.lstDiag] \  
   , "PatientTreatmentsList": [self.lstDrugs] , "drugDrugInteractionJson": [Interactions] })

## 4. Hive Table Structure, Hive db

Receiptarchive database

ReceiptHistory table

-- hive

create if not exist database receiptarchive;

CREATE EXTERNAL TABLE IF NOT EXISTS receiptarchive.ReceiptHistory(DoctorLicense String, PatientTZ String, PatientFirstName String,ConflictSExists Boolean, PatientLastName String, KupatHolim String, PtienBirthdate String, DrugName String, DrugId String, DrugDose String, dateCreated String, PatientDiseasesLqist String, PatientTreatmentsList String,

drugDrugInteractionJson String)

stored as Parquet

Location '/user/hive/warehouse/receiptarchive.db';

|  |  |  |
| --- | --- | --- |
| Field | Description | Possible use |
| DoctorLicense | Doctor License number |  |
| PatientTZ | Patient Personal id number |  |
| PatientFirstName |  |  |
| PatientLastName |  |  |
| ConflictSExists | If new receipt drug has conflicts with existing in health history treatment list | True or false |
| KupatHolim |  |  |
| PtienBirthdate |  |  |
| DrugName |  |  |
| DrugId |  |  |
| DrugDose |  |  |
| dateCreated |  |  |
| PatientDiseasesLqist | From patient health history |  |
| PatientTreatmentsList | From Patient health history |  |
| drugDrugInteractionJson | Interaction check result for this patient |  |

## 5. Mango db

**Db= DrugInteraction**

**Collection- Drugs**

if mycol.count\_documents({"MainDrugId": target\_df['MainDrugId'][0]}, limit=1) == 0:  
 post = {  
 "MainDrugId": target\_df['MainDrugId'][0],  
 "MainDrugName": target\_df['MainDrugName'][0],  
 "DrugsIdList": target\_df['DrugsIdList'],  
 "drugNameList": target\_df['drugNameList'],  
 "severityList": target\_df['severityList'],  
 "descriptionList": target\_df['descriptionList']  
 }

## 6. Lexical analysis and data transformation

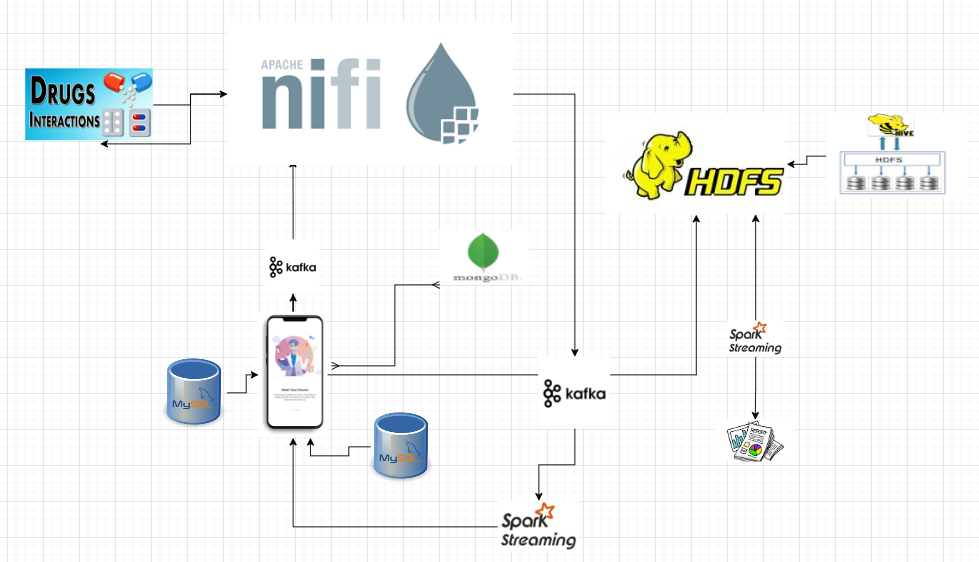
|  |  |  |
| --- | --- | --- |
| **Word** | **interaction** | **severity** |
| risk | The risk or severity of …increased | -3 |
| excretion rate | decrease the excretion rate … higher …level | -1 |
| metabolism | The metabolism … decreased | -1 |
| therapeutic efficacy | therapeutic efficacy … decreased | -1 |
| activities | decrease…activities | -2 |
| **adverse effects** | **risk or severity of adverse effects can be increased** | **-5** |
| metabolism | metabolism …increased | 1 |
| therapeutic efficacy | therapeutic efficacy …increased | 3 |
| increase | increase …activities | 2 |

Implement by regular expression

def ClassifyDesc(desc):  
 res=0  
 if bool(re.search("adverse effects.\*increased", desc)):  
 res= -5  
 elif bool(re.search("therapeutic efficacy.\*increased", desc)):  
 res=3  
 elif bool(re.search("therapeutic efficacy.\*decreased", desc)) :  
 res=-2  
 elif bool(re.search("risk or severity.\*increased", desc)) :  
 res=-4  
 elif bool(re.search("excretion rate.\*decreased", desc)):  
 res=-1  
 elif bool(re.search("metabolism.\*increased", desc)) :  
 res = 1  
 elif bool(re.search("metabolism.\*decreased", desc)) :  
 res = -1  
 elif bool(re.search("increase.\*activities", desc)):  
 res = 2  
 elif bool(re.search("decrease.\*activities", desc)):  
 res = -2  
 else:  
 res=0  
 #print (res)  
 return res



# **Architecture implemented getDrug Interaction**



# **Possible reports and queries**

* Report –amount of receipts with side effects and risks, which part of receipts cause negative effects to patients
* Which doctor gives more drugs with serious side effects and/or risks
* Which diseases are more likely to conflict with the drugs taken for other diseases and require increased doctor's caution

Etc

select Count(\*), DoctorLicense, KupatHolim

from receiptarchive.ReceiptHistory

where ConflictSExists

group by DoctorLicense, KupatHolim

# **Scripts examples**

**Producer**

|  |
| --- |
| import configuration as c from kafka import KafkaProducer from time import sleep  class Producer():  def \_\_init\_\_(self):  self.producer = KafkaProducer(bootstrap\_servers=c.bootstrapServers)  # self.request = []   def send(self, topic,req):  self.producer.send(topic, value=req.encode('utf-8') )    self.producer.flush()  self.producer.close() |

**Consumer**

ef checkDrugKf(drugName,Pat1):  
 try:  
 pr1 = pr.Producer()  
 lstSeverityUsedIter = []  
 lstDescUsedIter = []  
 topic = c.topic2 + drugName.capitalize()  
 consumer = KafkaConsumer(topic, bootstrap\_servers=c.bootstrapServers)  
 pr1.send(c.topic1,drugName.capitalize())  
 print("before")  
 for message in consumer:  
 print("into")  
 valJson=json.loads(message.value)  
 paDf=pd.json\_normalize(valJson) #, record\_path =['students'])  
 #Medcine check interaction  
 # exclude duplicates  
 id= paDf['rxcui'][0]  
 name=paDf['name'][0]  
 idList=[di for di in paDf['IdList'][0] if di != id]  
 NameList=[ni for ni in paDf['NameList'][0] if ni != name]  
 descLst=paDf['description'][0]  
 severityList=[ClassifyDesc(desc) for desc in descLst]  
 dictInterFull = {'MainDrugId':id,"MainDrugName":name,"DrugsIdList": idList,'drugNameList': NameList, 'severityList': severityList, 'descriptionList': descLst}  
 write\_df\_mongo(dictInterFull)  
  
 lstUsedInter=[el for el in NameList if el in Pat1.ConstantDrugsList]  
 lstIndexUsedIter=[NameList.index(el) for el in NameList if el in Pat1.ConstantDrugsList]  
 if len(lstIndexUsedIter)>0:  
 for i in lstIndexUsedIter:  
 lstSeverityUsedIter.append(severityList[i])  
 lstDescUsedIter.append(descLst[i])  
  
 #print (lstSeverityUsedIter,lstDescUsedIter)  
 break  
 dict = {'drug name': lstUsedInter, 'severity': lstSeverityUsedIter, 'description': lstDescUsedIter}  
 resDf=pd.DataFrame(dict)  
 print (resDf)  
 print(id)  
 return id,resDf  
 except Exception as e:  
 print(e)  
 return pd.DataFrame()

This script is for saving a request data, as is, on HDFS as JSON file for Archive and load it to hive.

|  |
| --- |
| def crReceipt(self):  # save to hdfs all data for receipt for followed analysis  try:  Interactions= self.DfInteraction.to\_json()  # print (Interactions)  # print(type(Interactions) )  #### Generating Today's date ####  today = date.today()  current\_date = today.strftime("%d\_%m\_%Y")  currDT = today.strftime( "%m/%d/%Y, %H:%M:%S")  ConflictSExists=False  for i in self.DfInteraction["Severity"]:  if int(i)>0:  continue  else:  ConflictSExists=True  break     df =pd.DataFrame(data={"DoctorLicense" : [self.DoctorL], "PatientTZ": [self.PatientTz], "PatientFirstName": [self.PatientFName]\  , "ConflictSExists": ConflictSExists,"PatientLastName": [self.PatientLName], "KupatHolim": [self.PatientKH], "PtienBirthdate": [self.PatientBD] \  , "DrugName": [self.DrugName] , "DrugId": [self.DrugId] ,"DrugDose": [self.DrugDose] , "dateCreated": [currDT] \  ,"PatientDiseasesLqist": [self.lstDiag] \  , "PatientTreatmentsList": [self.lstDrugs] , "drugDrugInteractionJson": [Interactions] })     path\_tbl = "Receipt"+ self.PatientTz + self.DrugName+ current\_date  # print (path\_tbl)  fs = pa.hdfs.connect(host='Cnt7-naya-cdh63', port=8020, user='hdfs', kerb\_ticket=None, extra\_conf=None)  #2. Create/clean the staging folder in HDFS   if not (fs.exists(c.hdfs\_json\_path)):  fs.mkdir(c.hdfs\_json\_path)  # print(f"{c.hdfs\_json\_path} created")  df\_for\_hdfs = pa.Table.from\_pandas(df)   with fs.open(c.hdfs\_json\_path + path\_tbl, "wb") as fw:  pq.write\_table(df\_for\_hdfs, fw)  fs.close()  #load to hive   hive\_cnx = hive.Connection(host=c.hdfs\_host,  port=c.hive\_port,  username=c.hive\_username,  password=c.hive\_password,  auth=c.hive\_mode)   cursor1 = hive\_cnx.cursor()  #for vendorid, TripStartMonth in part\_list:  strLoad = "LOAD DATA INPATH '" + c.hdfs\_json\_path + path\_tbl + "' INTO TABLE receiptarchive.ReceiptHistory"  # cursor1.execute('''LOAD DATA INPATH 'hdfs://Cnt7-naya-cdh63:8020/user/alin/de\_proj/traffic\_parquet/vendorid={0}/TripStartMonth={1}'  # INTO TABLE taxi.traffic  # PARTITION (vendorid={0}, TripStartMonth={1}) '''.format(vendorid, TripStartMonth))  print(strLoad)  cursor1.execute(strLoad)  cursor1.close()  hive\_cnx.commit()  hive\_cnx.close()      except Exception as e:  PatientClass.WriteLog("crReceipt error " +e, "Error", self.PatientTz)  print(e) |

This script is reading data from MySql

|  |
| --- |
| def findPatient(self):  # doctors.patients  try:  cnMS = mc.connect(  user=c.mysql\_username,  password=c.mysql\_password,  host=c.mysql\_host,  port=c.mysql\_port,  autocommit=True,  database=c.mysql\_database\_name)  cursor = cnMS.cursor()  com1 = "select FName,LNAme ,Bithdate,Deathdate ,ADress ,Mobile , Doctor , " \  + "KypatHolim ,DisesesList ,DrugsTreatmentList,Contraindications from patients where tz='" + self.TZ + "';"  #print (com1)   # WriteLog("Patient.findPatient sql=" + com1.replace("'",""), "Info",self.TZ, cnMS)  cursor.execute(com1)  if cursor.with\_rows:  WriteLog("Patient.findPatient row found=" , "Info",self.TZ)  row = cursor.fetchone() #.fetchall()  #print (row)  if row[8] != None:  self.DiseasesHistory =row[8].split(",")  #print(type(row[10]))  #row= [t[0] for t in rows]  if row[9]!= None:  self.ConstantDrugsList=row[9].split(",")  if row[10]!= None:  self.KnownPersonalDrugConflictsList=row[10].split(",")   self.Birthdate=row[2]  self.FirstName=row[0]  self.LastName=row[1]  self.KupatHolim=row[7]  self.Doctor = row[6]  else:  WriteLog("Patient.findPatient row not found=", "Info", self.TZ)  cursor.close()  cnMS.close()  except Exception as e:  WriteLog("Patient.findPatient error:"+ e, "Error", self.TZ) |

Write To MySql

def addDiagnose (self,diagnose):  
 # add new diagnose to Patient  
 try:  
 cnMS = mc.connect(  
 user=c.mysql\_username,  
 password=c.mysql\_password,  
 host=c.mysql\_host,  
 port=c.mysql\_port,  
 autocommit=True,  
 database=c.mysql\_database\_name)  
 if diagnose.lower() in self.DiseasesHistory:  
 WriteLog("the diagnosis already exist in patient history" + diagnose.lower() , "Info", self.TZ )  
 exit  
 cursor = cnMS.cursor()  
 self.DiseasesHistory.append(diagnose.lower())  
 #print (self.DiseasesHistory)  
 #print (",".join(self.DiseasesHistory))  
 sql = "update patients set DisesesList='" + ",".join(self.DiseasesHistory) +"' where tz='" + self.TZ + "';"  
 #print (sql)  
 WriteLog("Patient addDiagnose sql=" + sql.replace("'","") , "Info",self.TZ)  
 cursor.execute(sql)  
 #print(cursor.rowcount, "record(s) affected")  
 WriteLog("Patient addDiagnose rows affected=" + str(cursor.rowcount) , "Info",self.TZ)  
 cnMS.commit()  
 cursor.close()  
 cnMS.close  
 except Exception as e:  
 WriteLog("Patient.addDiagnose error:"+ e, "Error", self.TZ)

**Save to Mango**

This script is saving a filtered data to mango db for future use

def write\_df\_mongo(target\_df):  
 try:  
 mogodb\_client = pymongo.MongoClient('mongodb://localhost:27017/')  
 mydb = mogodb\_client["DrugInteraction"]  
 mycol = mydb["Drugs"]  
  
  
 if mycol.count\_documents({"MainDrugId": target\_df['MainDrugId'][0]}, limit=1) == 0:  
 post = {  
 "MainDrugId": target\_df['MainDrugId'][0],  
 "MainDrugName": target\_df['MainDrugName'][0],  
 "DrugsIdList": target\_df['DrugsIdList'],  
 "drugNameList": target\_df['drugNameList'],  
 "severityList": target\_df['severityList'],  
 "descriptionList": target\_df['descriptionList']  
 }  
 mycol.insert\_one(post)  
 print('item inserted')  
 else:  
 # myquery = {"MainDrugId":target\_df['MainDrugId'][0]}  
 # newvalues = {"$set": {"DrugsIdList": target\_df['DrugsIdList'],  
 # "drugNameList": target\_df['drugNameList'],  
 # "severityList": target\_df['severityList'],  
 # "descriptionList": target\_df['descriptionList']}}  
 #  
 # mycol.update\_one(myquery, newvalues)  
  
 print("already exist")  
  
 except Exception as e:  
 print(e)

**Create Database, Tables mysql**

|  |
| --- |
| # Create Database:  CREATE DATABASE IF NOT EXISTS doctors;  # Create Tables:  CREATE TABLE `patients` (  `tz` varchar(15) NOT NULL,  `FName` varchar(20) DEFAULT NULL,  `LNAme` varchar(40) DEFAULT NULL,  `Bithdate` date DEFAULT NULL,  `Deathdate` date DEFAULT NULL,  `ADress` varchar(100) DEFAULT NULL,  `Mobile` varchar(16) DEFAULT NULL,  `Doctor` varchar(15) DEFAULT NULL,  `KypatHolim` varchar(15) DEFAULT NULL,  `DisesesList` varchar(1000) DEFAULT NULL,  `DrugsTreatmentList` varchar(1000) DEFAULT NULL,  `Contraindications` varchar(500) DEFAULT NULL  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;  create table doctors( tz varchar(15),  licienNum varchar(40),  FName varchar(20),  LNAme varchar(40),  specialisation varchar(50),  workPlaceList varchar (500)  );  CREATE TABLE `Doctor\_Assistance\_App\_Error\_Log` (  `LogId` int(11) NOT NULL AUTO\_INCREMENT,  `Event\_Datetime` datetime DEFAULT CURRENT\_TIMESTAMP,  `doctorId` varchar(40) DEFAULT NULL,  `PatientTz` varchar(15) DEFAULT NULL,  `MsgType` varchar(50) DEFAULT NULL,  `Msg\_text` varchar(1000) DEFAULT NULL,  PRIMARY KEY (`LogId`)  ) ENGINE=InnoDB AUTO\_INCREMENT=61 DEFAULT CHARSET=latin1;  Create table receipt (  DoctorLic varchar(40),  PatientTZ varchar(15),  PatientFName varchar(20),  PatientLName varchar(40),  PatientBD date,  PatientKH varchar(15)  DrugId varchar(15),  DrugName varchar(50)  DrugDose varchar(50)  DateCreated datetime);  ) |
|  |
|  |