UMKC

School of Computing
and Engineering

Dynamic Intelligent Q/A System

PROJECT INCREMENT 1 REPORT

SUMMER 2017

Team – 3

Nageswara, Rao Nandigam – 18
Chakilam, Revanth – 2

Syed, Moin – 28

Sankar, Pentyala – 22

Motivation:

We the Team Innovators 2.0 are in a search of data and knowledge. But Importantly we have a lot of difference between information, data, statistics and knowledge. When we speak about Information intruding and a semantic google search is more improvised and we can retrieve relevant information sitting in home. "Question Answering" is a dynamic way of retrieving Information, which learns knowledge. The main focus is particular in obtaining the respective documents but also particular in obtaining the respective response to query we post. Question and Answer system is capable of performing the NLP Processing, Information gathering, topic discovery, Machine Learning and Sematic Search. Question and answer system itself is the beauty of NLP and same instance have a bit of science in its essence. Question and Answer System is required every aspect, let be in the field of health and sciences, an intelligent learned system for children at schools, professional assistant etc. So, this is necessary in each case when we require some help from computer as well. It goes without saying that it is worth exploring the exciting field of question answering.

Objective:

Our Project critically deals with the building of typical model of information knowledge retrieval, named Question & Answering model. If suppose any given query asked in natural human language, Our question and answer system is implemented in such a way to extract the reality possible answer in the form of a pre-defined named-entity type, that is a human person, or may be an organization, a location, etc. Thus, we are connecting the question objects with live entities in a given radius is important in question and answer system. The projects main motto is to enhance the performance of the Question Answering system by using the knowledge and data from the natural grouping of word in the document files.

Significance:

We are constructing a knowledge graph such a way to build the question and answering system to deliver answers very effectively. To give better the results from the system designed we are applying different techniques such as NLP operations, Information retrieval, topic discovery and knowledge discovery.

Q/A System:

For this project, we have taken the Data set from BBC sports concentrating on the sport Cricket. From this Data set we try to construct knowledge graph and making system dynamic to answer all possible questions on sports questions.

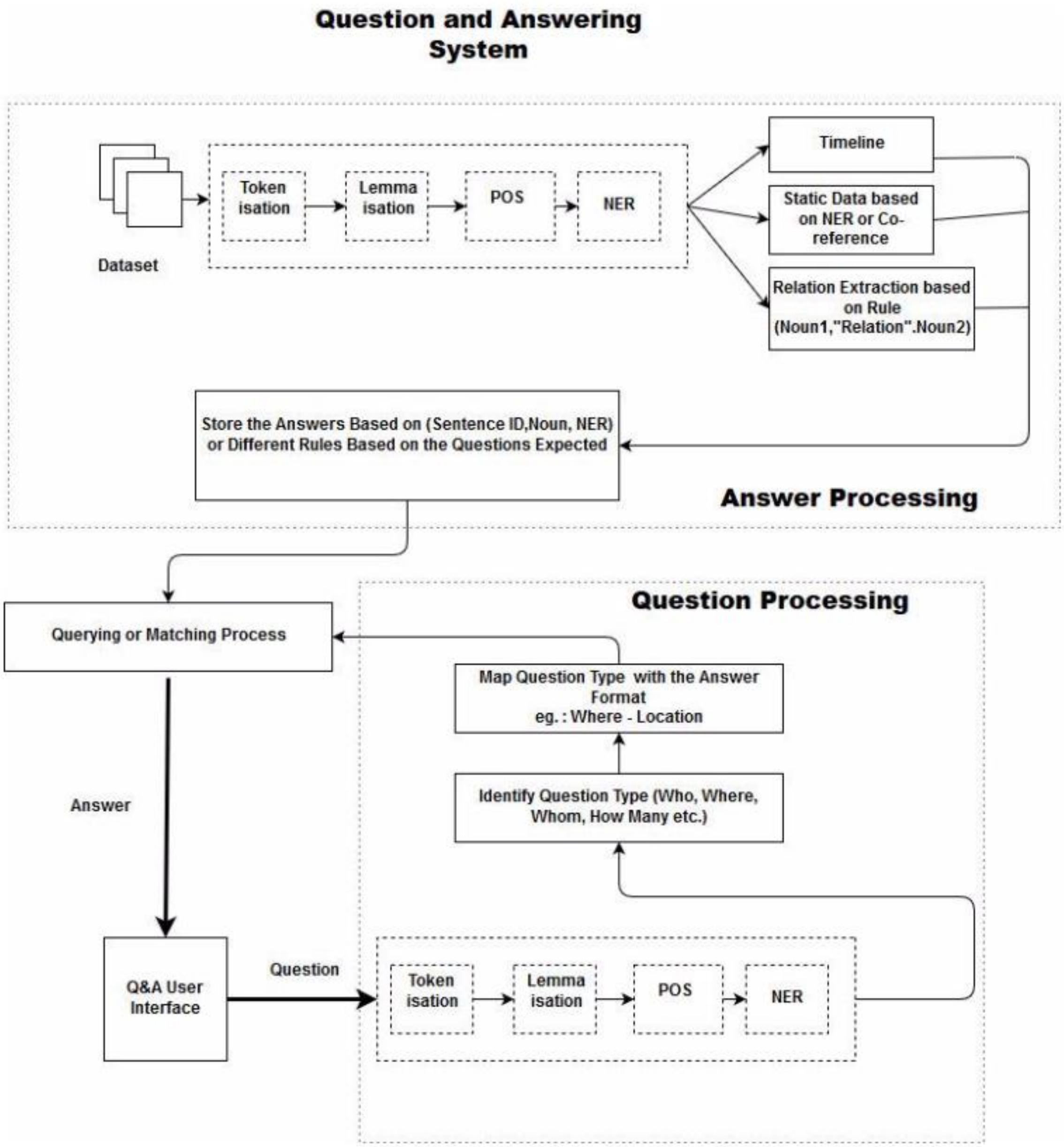
Datasets:

BBC Sports domain

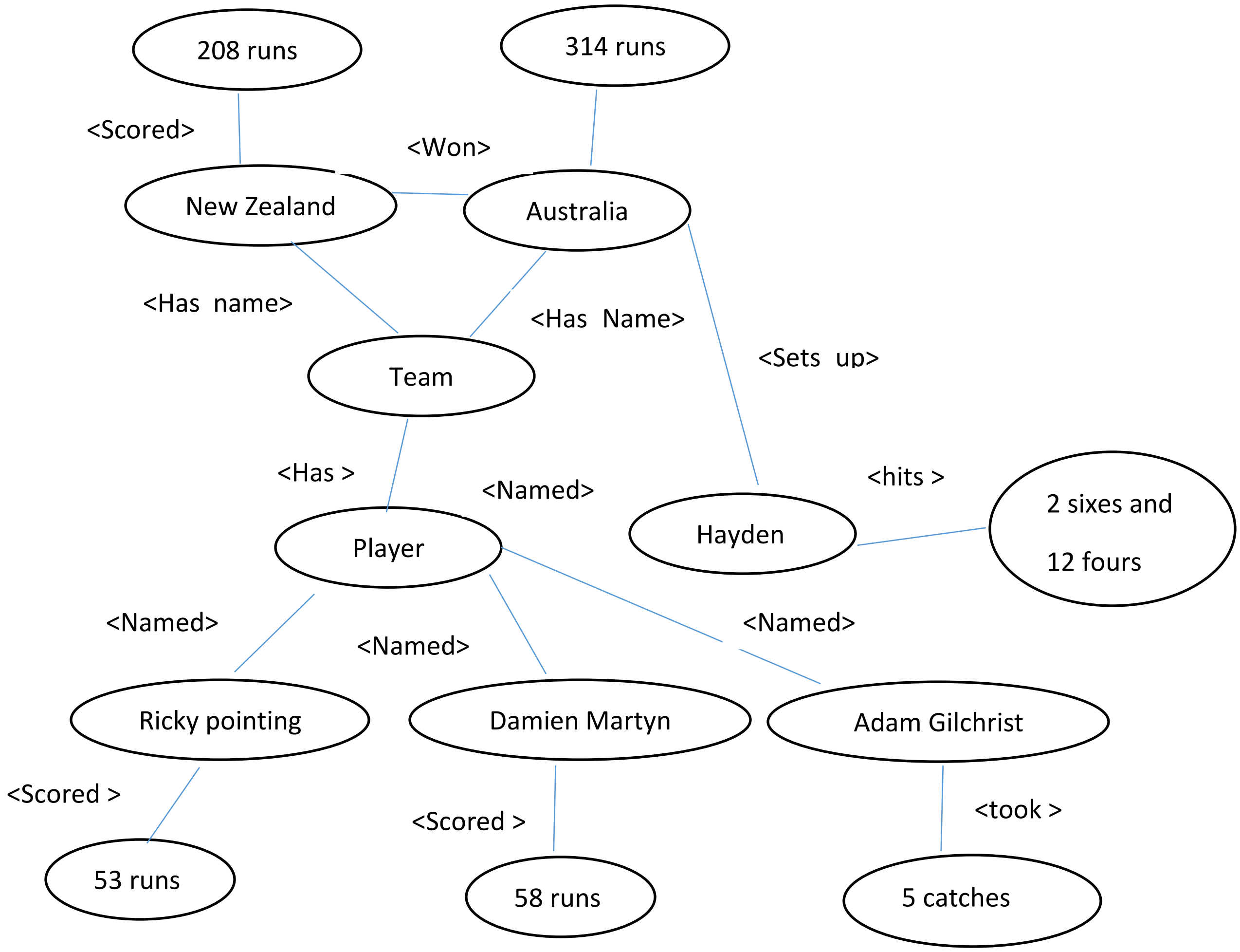
http://mlg.ucd.ie/datasets/bbc.html

Design:

Workflow:


Question and Answering System
Dataset
Token isation
Lemma isation
POS
NER
Timeline
Static Data based on NER or Co-reference
Relation Extraction based on Rule (Noun1,"Relation".Noun2)
Store the Answers Based on (Sentence ID,Noun, NER) or Different Rules Based on the Questions Expected
Answer Processing
Querying or Matching Process
Question Processing
Map Question Type with the Answer Format eg. : Where - Location
Identify Question Type (Who, Where, Whom, How Many etc.)
Answer
Q&A User Interface
Question
Token isation
Lemma isation
POS
NER

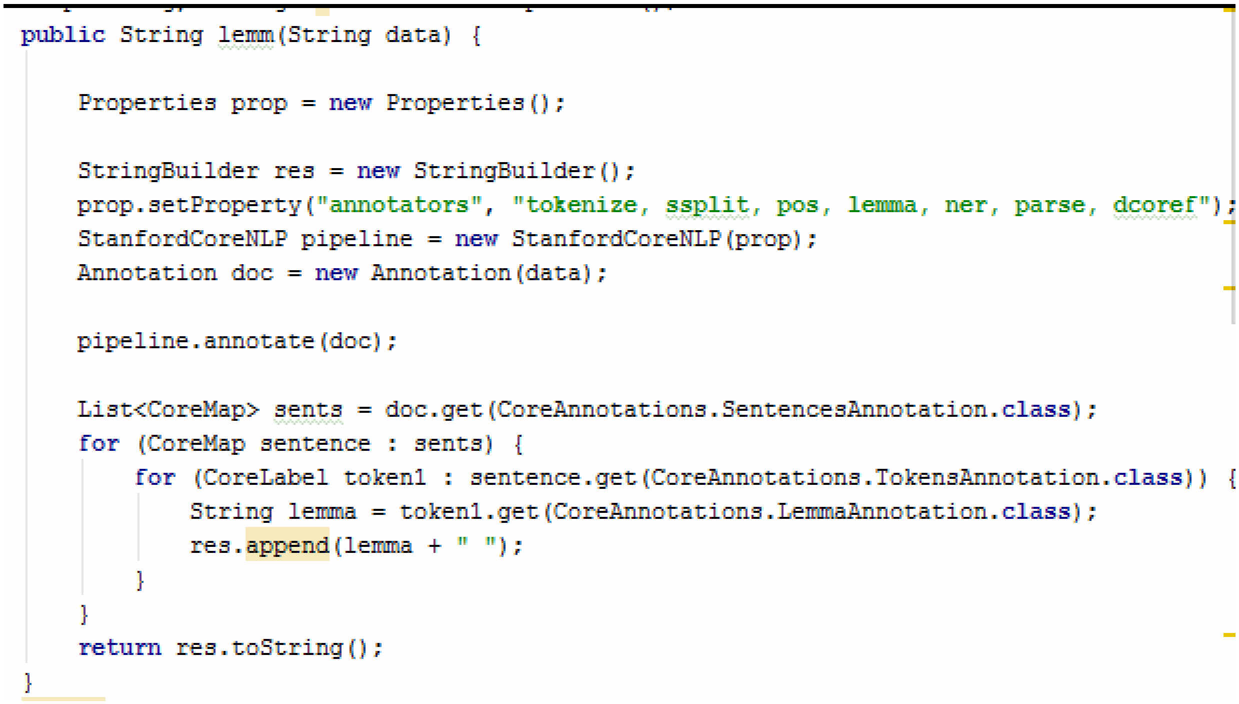
Knowledge Graph:



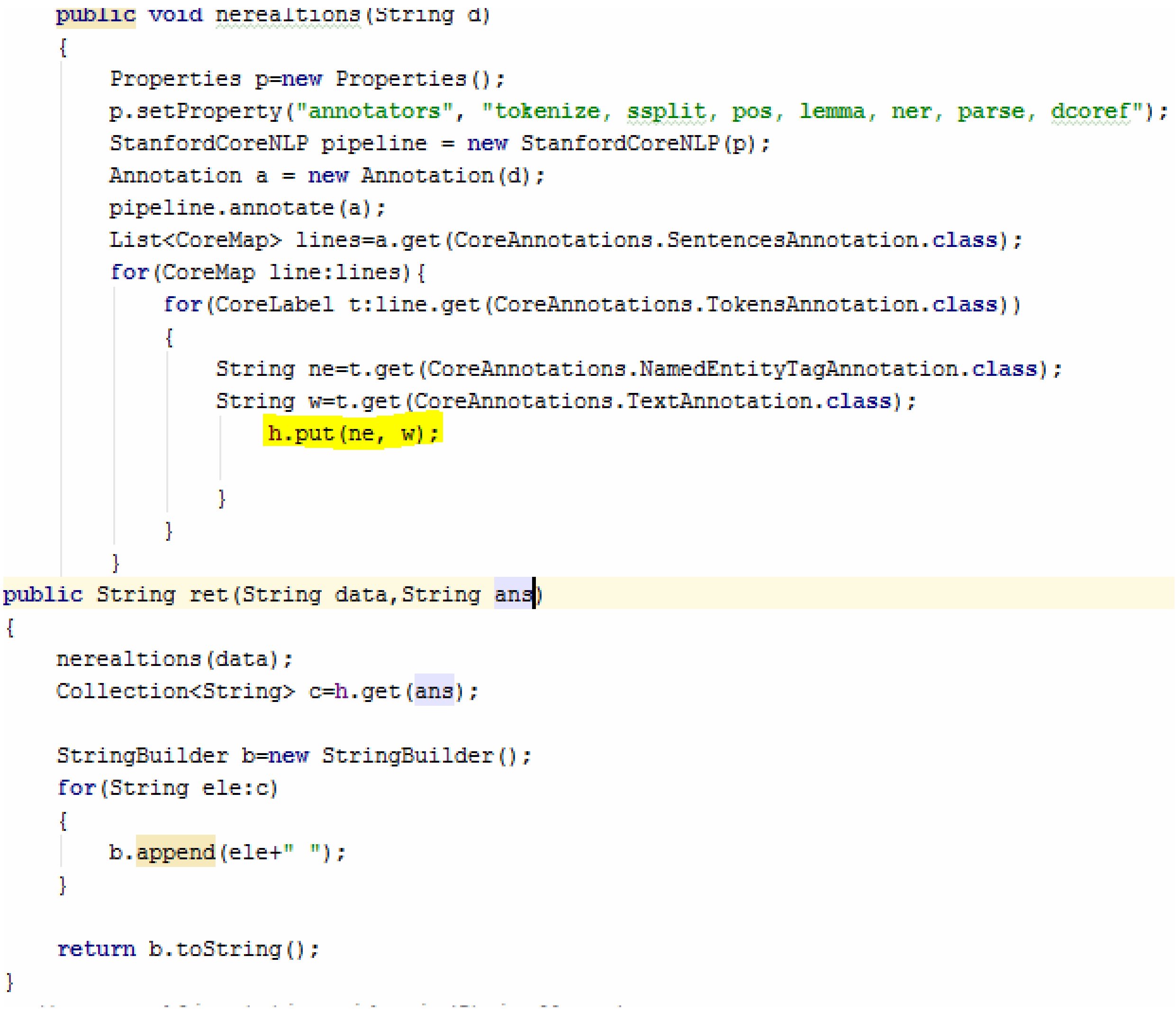
Implementation:

NLP Process:

1) Doing NLP processing of tokenizing, lemmatization and extracting named entity relations and storing it in HashMap.

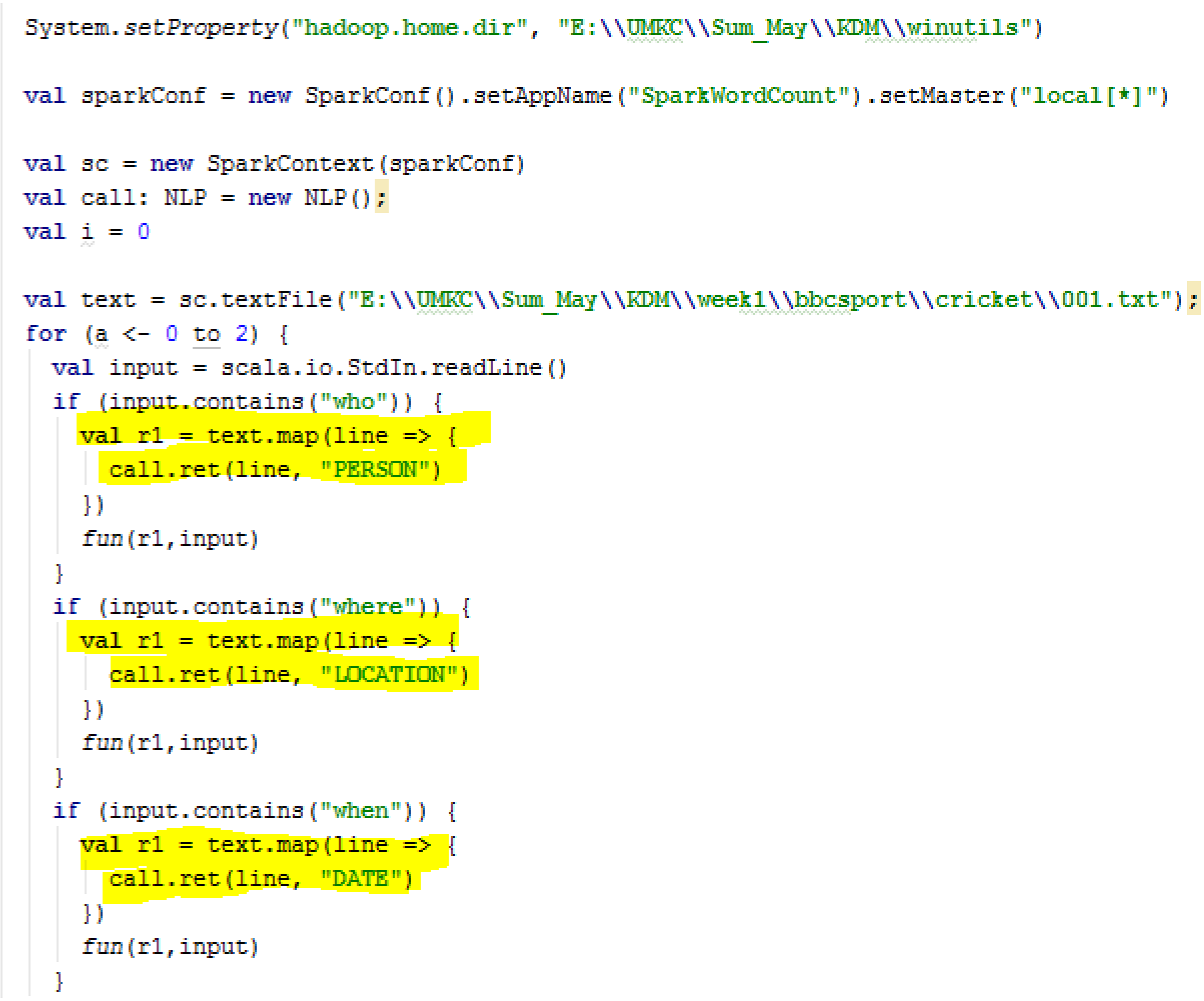


```
public String lemm(String data) {

    Properties prop = new Properties();

    StringBuilder res = new StringBuilder();
    prop.setProperty("annotators", "tokenize, ssplit, pos, lemma, ner, parse, dcoref");
    StanfordCoreNLP pipeline = new StanfordCoreNLP(prop);
    Annotation doc = new Annotation(data);

    pipeline.annotate(doc);

    List<CoreMap> sents = doc.get(CoreAnnotations.SentencesAnnotation.class);
    for (CoreMap sentence : sents) {
        for (CoreLabel token1 : sentence.get(CoreAnnotations.TokensAnnotation.class)) {
            String lemma = token1.get(CoreAnnotations.LemmaAnnotation.class);
            res.append(lemma + " ");
        }
    }
    return res.toString();
}
```



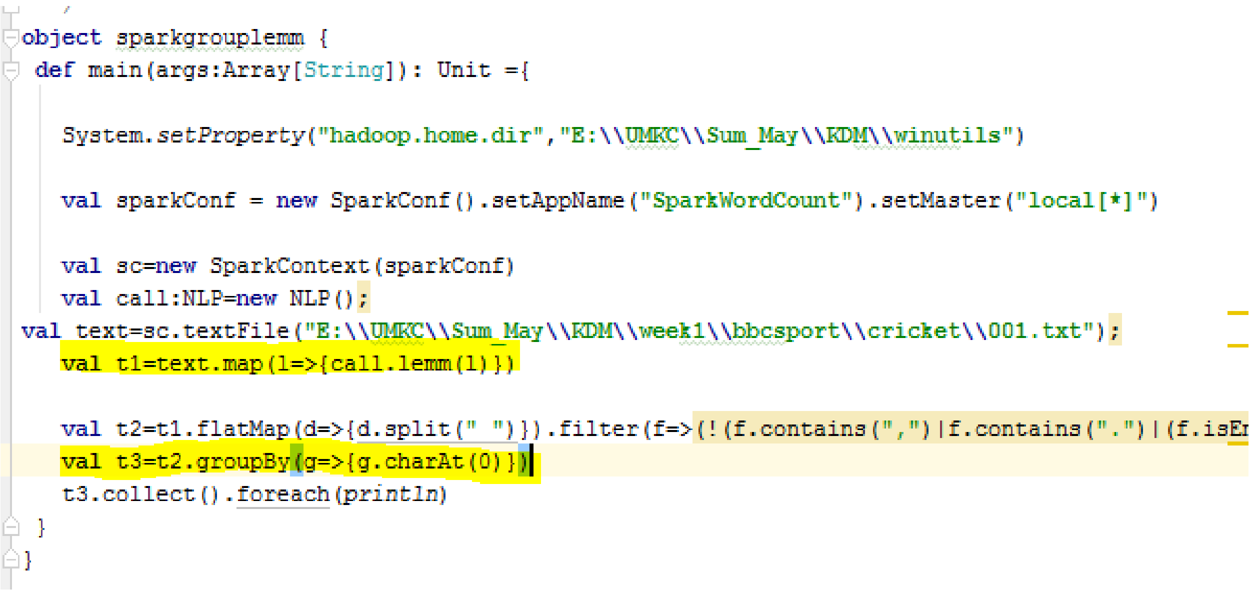
```
    public void nerealtions(String d)
    {
        Properties p=new Properties();
        p.setProperty("annotators", "tokenize, ssplit, pos, lemma, ner, parse, dcoref");
        StanfordCoreNLP pipeline = new StanfordCoreNLP(p);
        Annotation a = new Annotation(d);
        pipeline.annotate(a);
        List<CoreMap> lines=a.get(CoreAnnotations.SentencesAnnotation.class);
        for(CoreMap line:lines){
            for(CoreLabel t:line.get(CoreAnnotations.TokensAnnotation.class))
            {
                String ne=t.get(CoreAnnotations.NamedEntityTagAnnotation.class);
                String w=t.get(CoreAnnotations.TextAnnotation.class);
                    h.put(ne, w);

                }
            }
        }
public String ret(String data,String ans)
{
    nerealtions(data);
    Collection<String> c=h.get(ans);

    StringBuilder b=new StringBuilder();
    for(String ele:c)
    {
        b.append(ele+" ");
    }

    return b.toString();
}
```

Asking questions and finding answer type based on “wh” words



```
System.setProperty("hadoop.home.dir", "E:\\UMKC\\Sum_May\\KDM\\winutils")

val sparkConf = new SparkConf().setAppName("SparkWordCount").setMaster("local[*]")

val sc = new SparkContext(sparkConf)
val call: NLP = new NLP();
val i = 0

val text = sc.textFile("E:\\UMKC\\Sum_May\\KDM\\week1\\bbcsport\\cricket\\001.txt");
for (a <- 0 to 2) {
  val input = scala.io.StdIn.readLine()
  if (input.contains("who")) {
    val r1 = text.map(line => {
      call.ret(line, "PERSON")
    })
    fun(r1,input)
  }
  if (input.contains("where")) {
    val r1 = text.map(line => {
      call.ret(line, "LOCATION")
    })
    fun(r1,input)
  }
  if (input.contains("when")) {
    val r1 = text.map(line => {
      call.ret(line, "DATE")
    })
    fun(r1,input)
  }
```

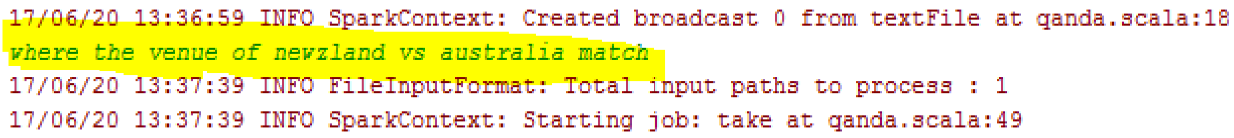
And based on answer type, we are calling particular document who is responsible to answer.



```
object sparkgrouplemm {
 def main(args:Array[String]): Unit ={

   System.setProperty("hadoop.home.dir","E:\\UMKC\\Sum_May\\KDM\\winutils")

   val sparkConf = new SparkConf().setAppName("SparkWordCount").setMaster("local[*]")

   val sc=new SparkContext(sparkConf)
   val call:NLP=new NLP();
val text=sc.textFile("E:\\UMKC\\Sum_May\\KDM\\week1\\bbcsport\\cricket\\001.txt");
   val t1=text.map(l=>{call.lemm(l)})

   val t2=t1.flatMap(d=>{d.split(" ")}).filter(f=>(!(f.contains(",")|f.contains(".")|(f.isEm
   val t3=t2.groupBy(g=>{g.charAt(0)})
   t3.collect().foreach(println)
 }
}
```

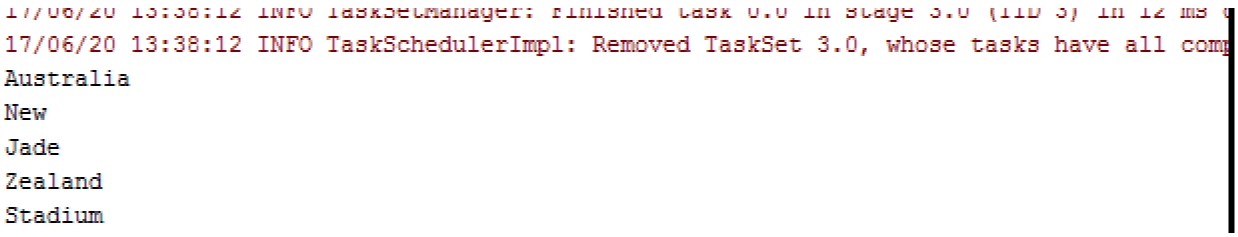
Question and answers:

Question 1:



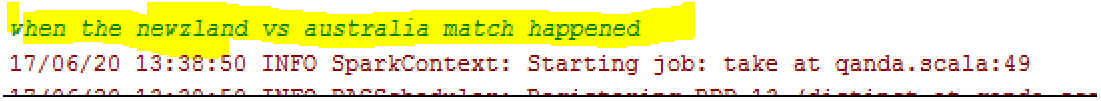
```
17/06/20 13:36:59 INFO SparkContext: Created broadcast 0 from textFile at qanda.scala:18
where the venue of newzland vs australia match
17/06/20 13:37:39 INFO FileInputFormat: Total input paths to process : 1
17/06/20 13:37:39 INFO SparkContext: Starting job: take at qanda.scala:49
```

Answer:



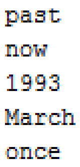
```
17/06/20 13:38:12 INFO TaskSetManager: Finished task 0.0 in stage 3.0 (TID 3) in 12 ms (
17/06/20 13:38:12 INFO TaskSchedulerImpl: Removed TaskSet 3.0, whose tasks have all comp
Australia
New
Jade
Zealand
Stadium
```

Question 2:



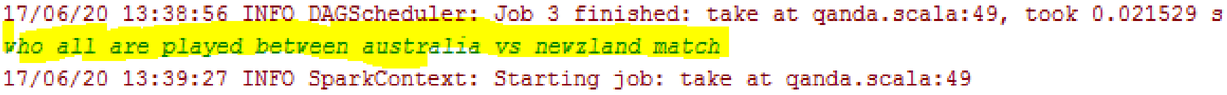
```
when the newzland vs australia match happened
17/06/20 13:38:50 INFO SparkContext: Starting job: take at qanda.scala:49
```

Answer:



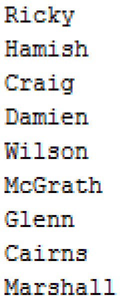
```
past
now
1993
March
once
```

Question 3:



```
17/06/20 13:38:56 INFO DAGScheduler: Job 3 finished: take at qanda.scala:49, took 0.021529 s
who all are played between australia vs newzland match
17/06/20 13:39:27 INFO SparkContext: Starting job: take at qanda.scala:49
```

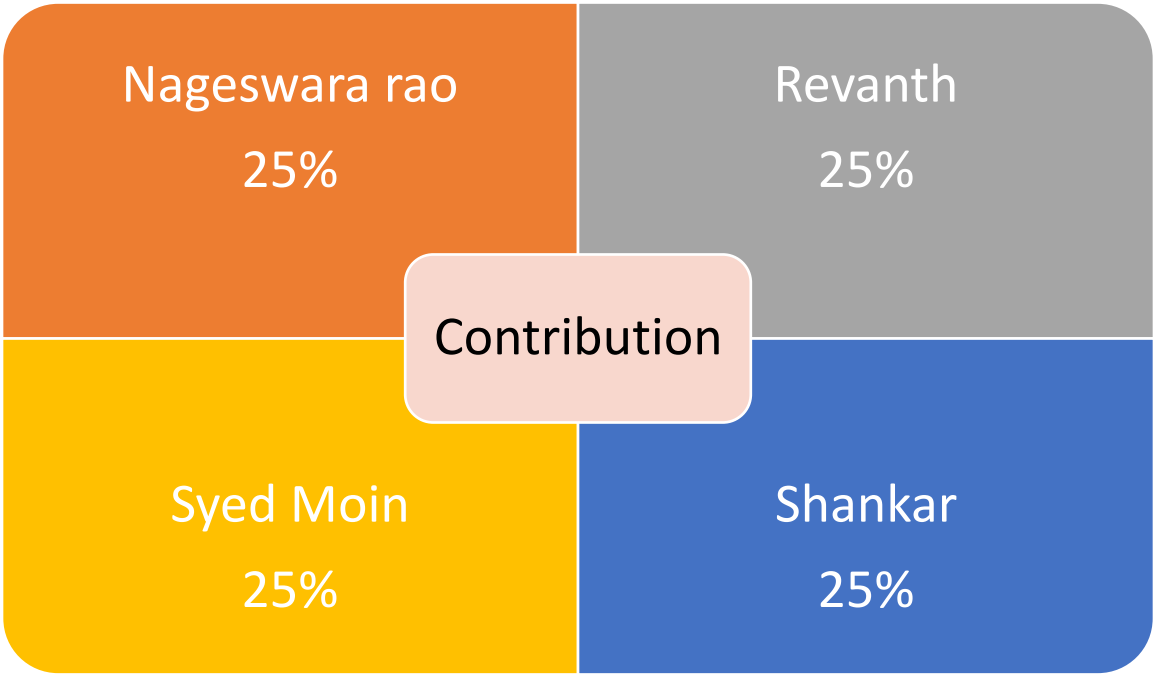
Answer:



```
Ricky
Hamish
Craig
Damien
Wilson
McGrath
Glenn
Cairns
Marshall
```

Project Management:

Contribution:



Nageswara rao

25%

Revanth

25%

Contribution

Syed Moin

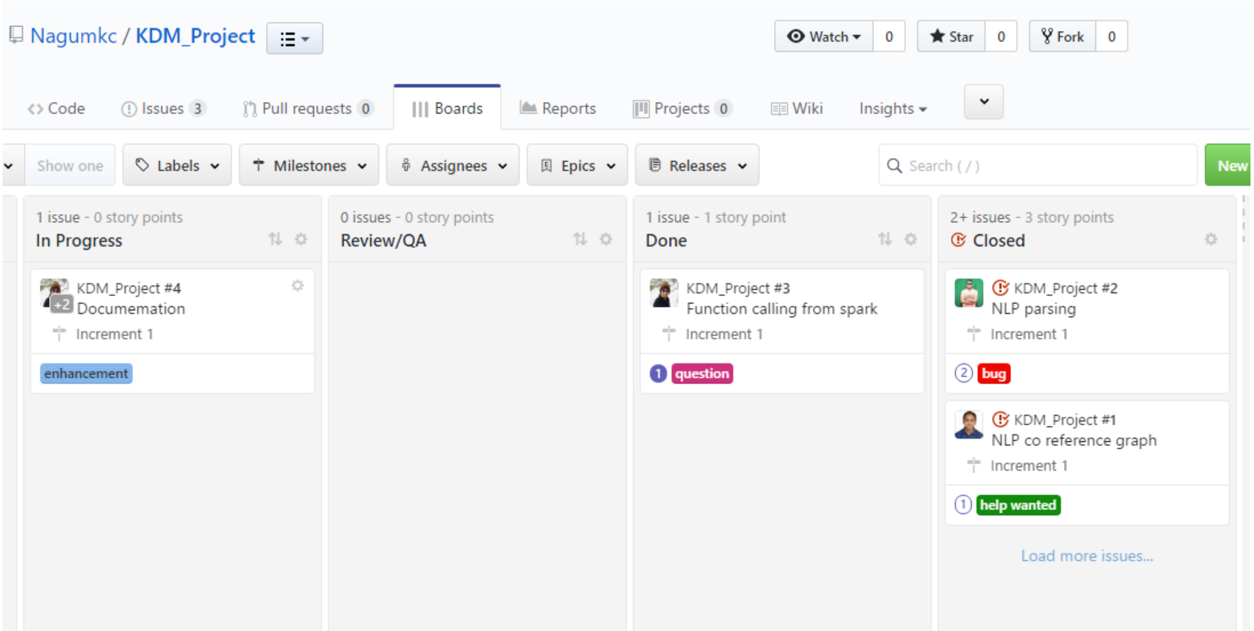
25%

Shankar

25%

GitHub screens:

Board:



1 issue - 0 story points
In Progress
- KDM_Project #4 Documemation — Increment 1 — enhancement

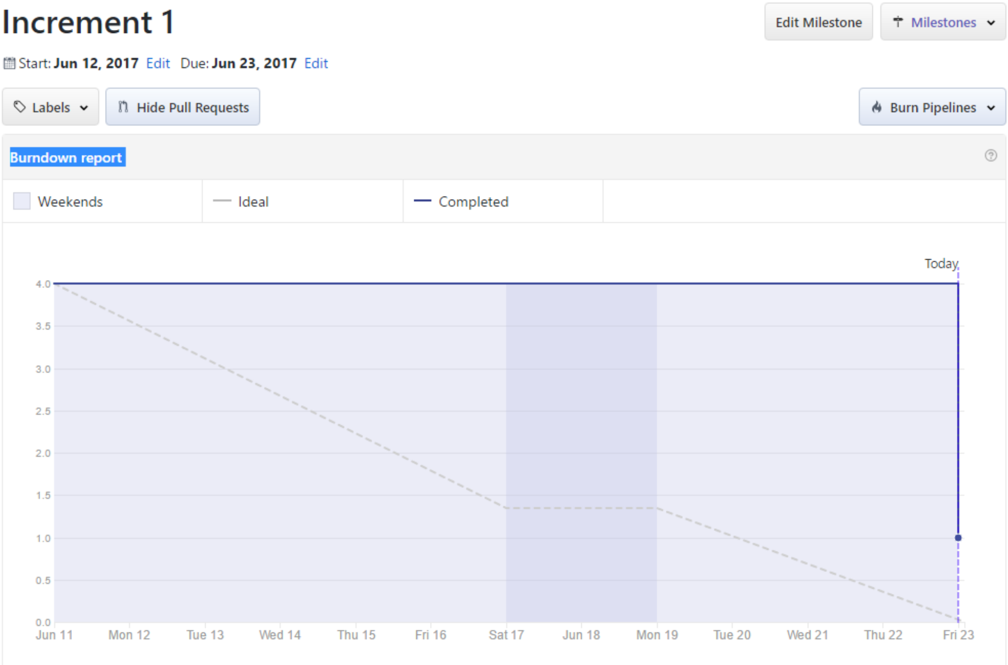
0 issues - 0 story points
Review/QA

1 issue - 1 story point
Done
- KDM_Project #3 Function calling from spark — Increment 1 — 1 question

2+ issues - 3 story points
Closed
- KDM_Project #2 NLP parsing — Increment 1 — 2 bug
- KDM_Project #1 NLP co reference graph — Increment 1 — 1 help wanted

Load more issues...

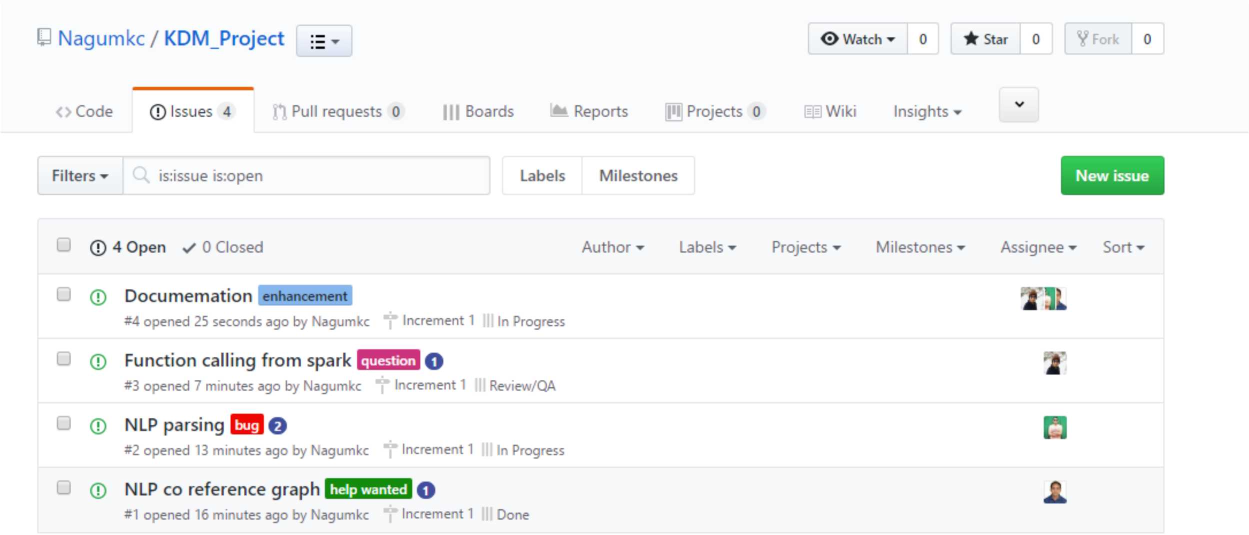
Burndown report:



Increment 1

Start: **Jun 12, 2017** Edit Due: **Jun 23, 2017** Edit

Issues:



Nagumkc / KDM_Project

4 Open ✓ 0 Closed

- Documemation enhancement
  #4 opened 25 seconds ago by Nagumkc Increment 1 In Progress
- Function calling from spark question 1
  #3 opened 7 minutes ago by Nagumkc Increment 1 Review/QA
- NLP parsing bug 2
  #2 opened 13 minutes ago by Nagumkc Increment 1 In Progress
- NLP co reference graph help wanted 1
  #1 opened 16 minutes ago by Nagumkc Increment 1 Done

Future work:

Till now, we have extracted the data statically from the Data set using the NLP techniques and have created question and answering system based on the extracted data.

The Future scope is to generate the Question and Answering based on the Knowledge graph dynamically by parsing data and finding out the entities and relationship between the entities. The main Entity extraction information is implicitly done using Natural language and explicitly done using structured data markup.