### Team Members :

#### Anmol More : 11915043

#### Dharani Kiran Kavuri : 11915033

#### Shubhendu Vimal : 11915067

#Ref : https://bnosac.github.io/udpipe/docs/doc7.html

shinyServer(function(input, output) {

options(shiny.maxRequestSize = 30 \* 1024 ^ 2)

dataset <- reactive({

if (is.null(input$text\_file)) {

return(NULL)

}

else {

doc = readLines(input$text\_file$datapath)

temp <- tolower(doc) #convert to lowercase

temp <-

stringr::str\_replace\_all(temp, "[^a-zA-Z\\s]", " ") #remove all non alphabets

temp <-

stringr::str\_replace\_all(temp, "[\\s]+", " ") #strip extra spaces

return(temp)

}

})

pos\_tags <- reactive({

pos\_tag\_names = list()

input\_pos\_tags <- as.vector(input$pos\_tags)

for (i in 1:length(input\_pos\_tags)) {

if (input\_pos\_tags[i] == 1) {

pos\_tag\_names[i] <- 'ADJ'

}

else if (input\_pos\_tags[i] == 2) {

pos\_tag\_names[i] <- 'NOUN'

}

else if (input\_pos\_tags[i] == 3) {

pos\_tag\_names[i] <- 'PROPN'

}

else if (input\_pos\_tags[i] == 4) {

pos\_tag\_names[i] <- 'ADV'

}

else if (input\_pos\_tags[i] == 5) {

pos\_tag\_names[i] <- 'VERB'

}

}

return(pos\_tag\_names)

})

#cols in annotated df :

#"doc\_id", "paragraph\_id", "sentence\_id", "sentence", "token\_id", "token", "lemma", "upos"

#"xpos", "feats", "head\_token\_id", "dep\_rel", "deps", "misc"

annotated\_df <- reactive({

model = udpipe\_load\_model("english-ewt-ud-2.4-190531.udpipe")

annotated\_text <-

udpipe\_annotate(model, x = dataset())

annotated\_df <- as.data.frame(annotated\_text)

write.csv(annotated\_df, 'annotated\_doc.csv')

return(annotated\_df)

})

#file for download

output$download\_df <- downloadHandler(

filename = function() {

"annotated\_doc.csv"

},

content = function(fname) {

write.csv(subset(annotated\_df(), select = -c(sentence)), fname)

}

)

#select top 100 rows to display

output$annotated\_table <- reactive({

display\_df <- subset(annotated\_df(), select = -c(sentence))

display\_top\_100 <- head(display\_df, 100)

return (renderDataTable(datatable(display\_top\_100)))

})

cooccurance\_df <- reactive({

cooccurance\_text <- cooccurrence(

x = subset(annotated\_df(), upos %in% pos\_tags()),

term = "lemma",

group = c("doc\_id", "paragraph\_id", "sentence\_id")

)

return(cooccurance\_text)

})

#select top 30 from at doc level

output$co\_occurance\_plot <- renderPlot({

wordnetwork <- head(cooccurance\_df(), 30)

wordnetwork <-

igraph::graph\_from\_data\_frame(wordnetwork)

ggraph(wordnetwork, layout = "fr") +

geom\_edge\_link(aes(width = cooc, edge\_alpha = cooc), edge\_colour = "blue") +

geom\_node\_text(aes(label = name), col = "black", size = 4) +

theme\_graph(base\_family = "Arial Narrow") +

labs(title = "Top 30 Cooccurrence graph",

subtitle = paste(pos\_tags(), collapse = ","))

})

#filter annotated doc by Noun

nouns\_by\_freq <- reactive({

noun\_doc = subset(annotated\_df(), upos == 'NOUN')

nouns\_by\_freq = txt\_freq(noun\_doc$lemma) #contains key, freq, freq\_pct sorted by freq

return(nouns\_by\_freq)

})

#get top 100 nouns plot

output$word\_cloud\_noun <- renderPlot({

top\_words <-

head(nouns\_by\_freq(), 200) #take 200 words, limit same in UI

wordcloud(

words = top\_words$key,

freq = top\_words$freq,

min.freq = input$min\_freq,

max.words = input$max\_words

)

})

#filter annotated doc by verb

verbs\_by\_freq <- reactive({

verb\_doc = subset(annotated\_df(), upos == 'VERB')

verbs\_by\_freq = txt\_freq(verb\_doc$lemma) #contains key, freq, freq\_pct sorted by freq

return(verbs\_by\_freq)

})

#get top 100 verbs plot

output$word\_cloud\_verb <- renderPlot({

top\_words <-

head(verbs\_by\_freq(), 200) #take 200 words, limit same in UI

wordcloud(

words = top\_words$key,

freq = top\_words$freq,

min.freq = input$min\_freq,

max.words = input$max\_words

)

})

})