

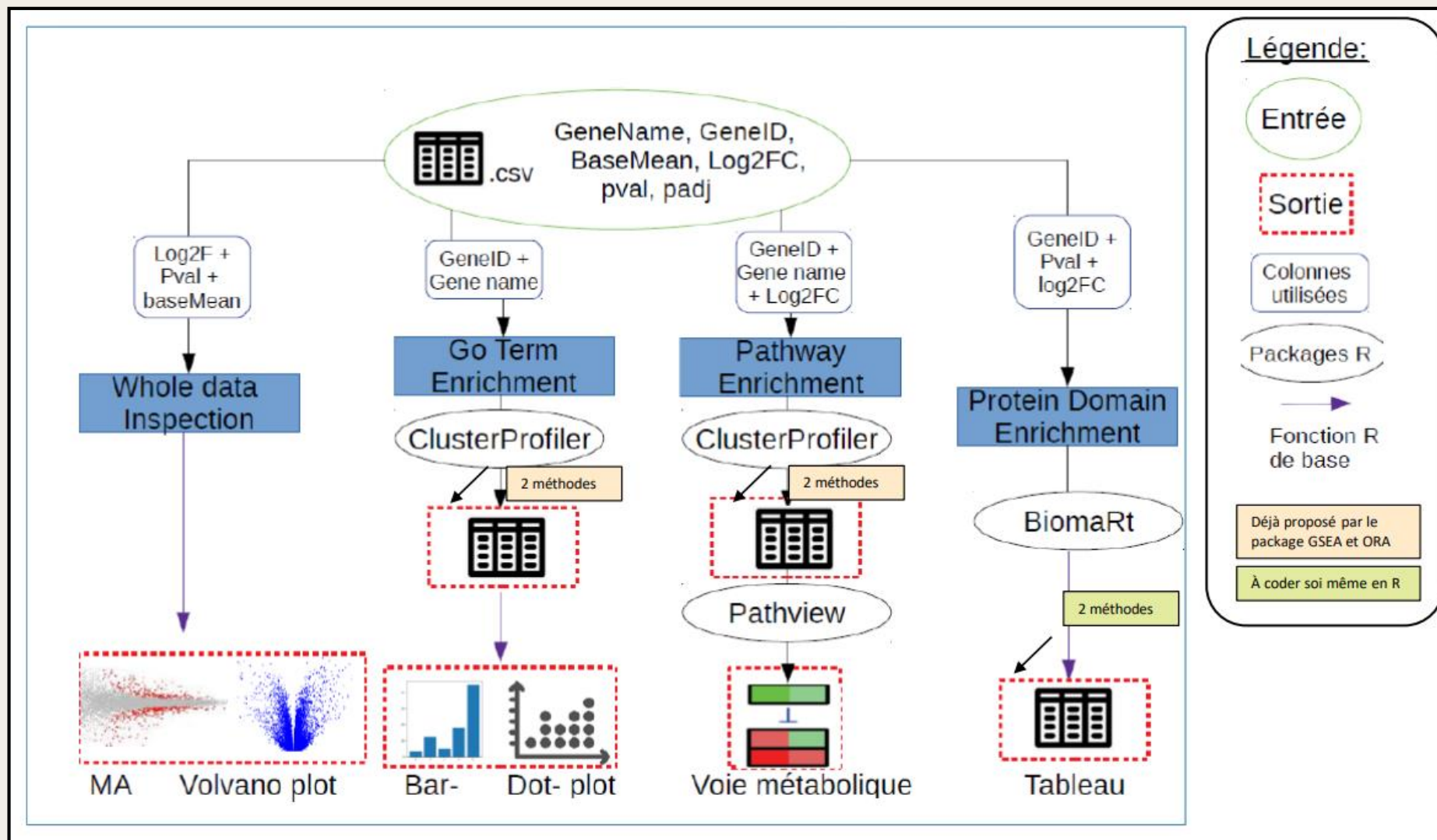
Projet d'Enrichissement Fonctionnel avec R Shiny

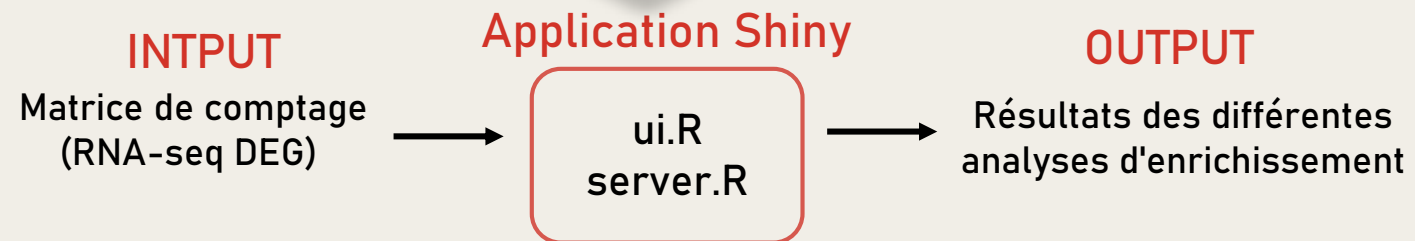
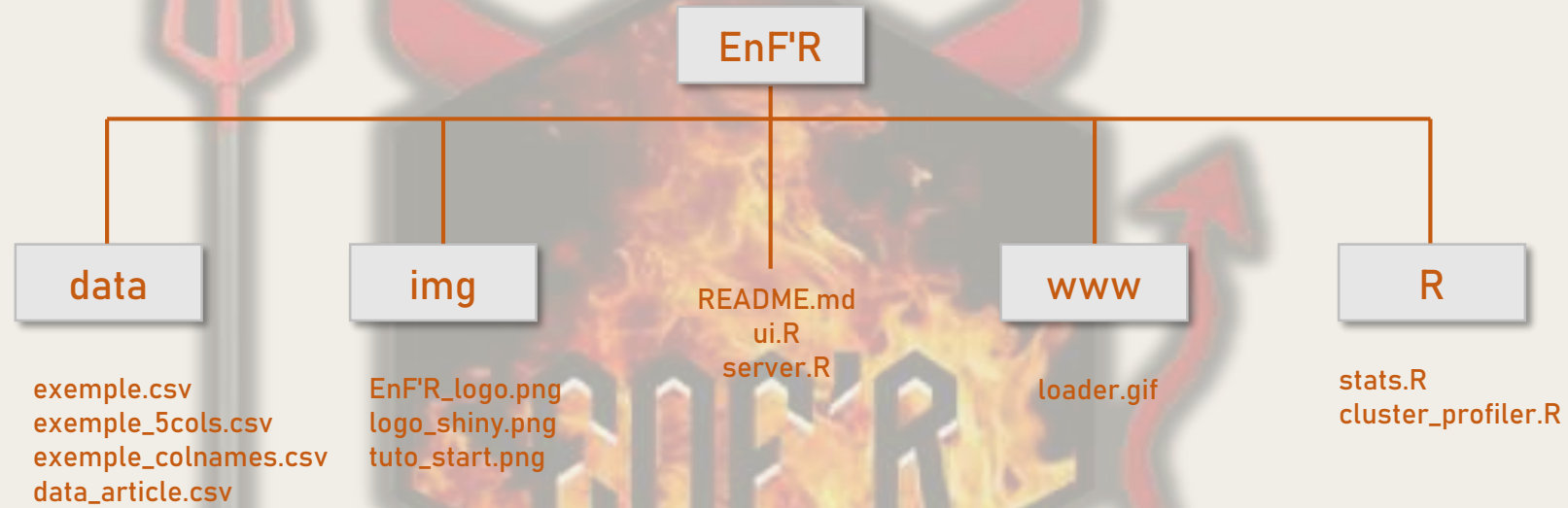
Solène PETY – Meije MATHE – Bryce LETERRIER – Louis OLLIVIER
Equipe EnF'R

UE 5 – Bioinformatique en sciences omiques 3
Master 2.1 mention Bioinformatique
Parcours BIMS
2021-2023

Université de Rouen-Normandie

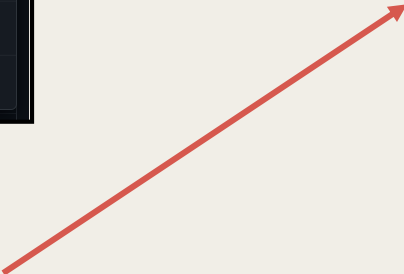

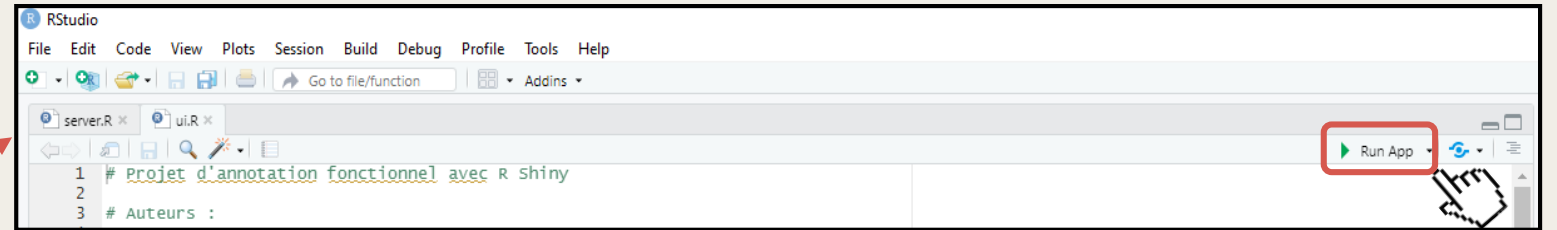
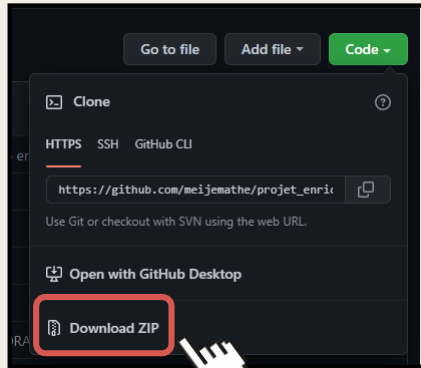







https://github.com/meijemathe/projet_enrichissement_fonctionnel.git

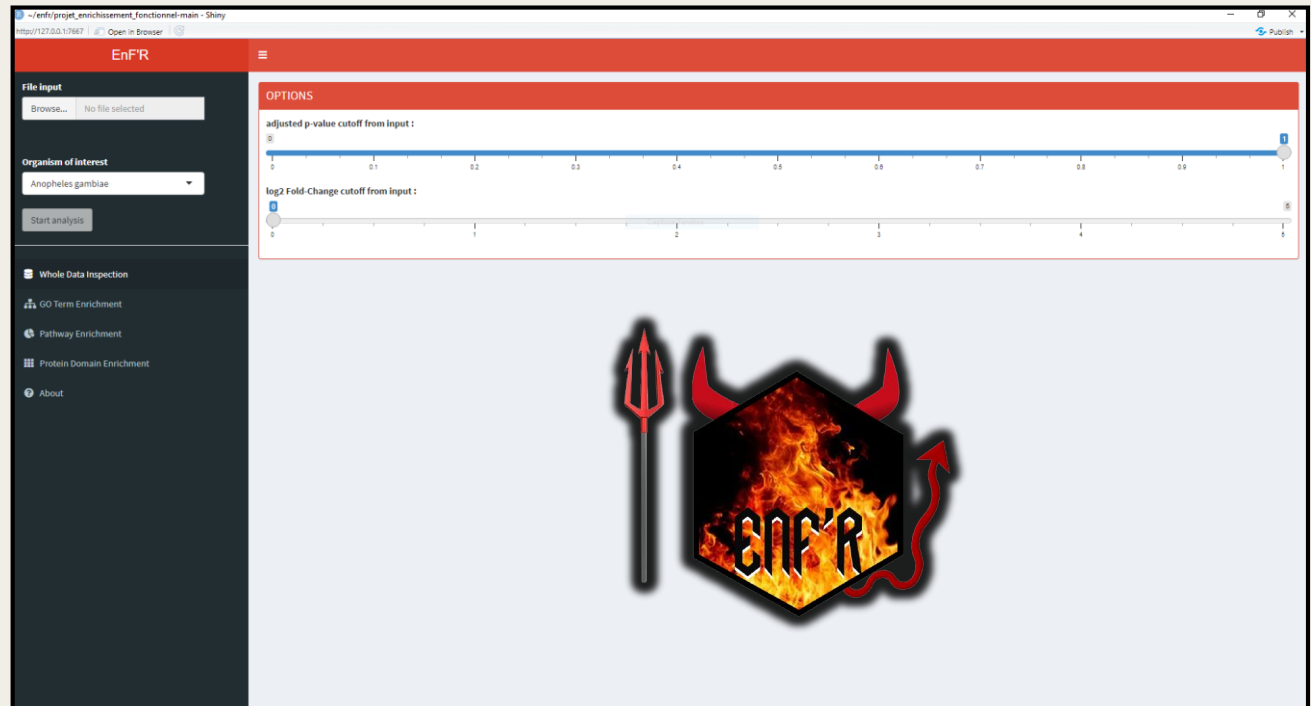




data	10/06/2022 15:18	Dossier de fichiers	
img	10/06/2022 15:18	Dossier de fichiers	
R	10/06/2022 15:18	Dossier de fichiers	
www	10/06/2022 15:18	Dossier de fichiers	
clusterProfiler.Rmd	10/06/2022 15:18	Fichier RMD	8 Ko
organigramme_readme	10/06/2022 15:18	Présentation Micr...	224 Ko
README.md	10/06/2022 15:18	Fichier MD	8 Ko
server	10/06/2022 15:18	Fichier R	21 Ko
ui	10/06/2022 15:18	Fichier R	25 Ko



Lancement de la démonstration

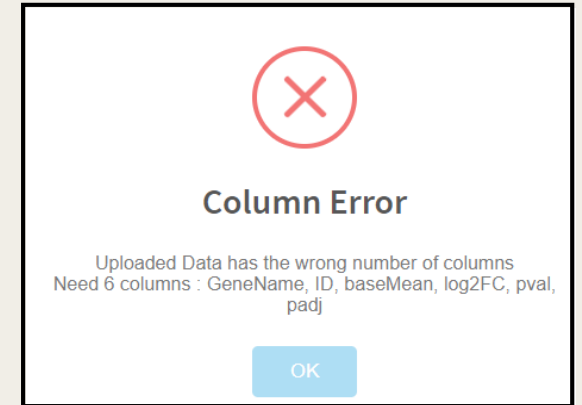


```
observeEvent(input$file, {
  req(input$file)
  #test extension
  ext <- tools::file_ext(input$file$datapath)
  validate(need(ext == "csv", "Please upload a csv file"))
  # Vector with good columns names
  good <- c("GeneName", "ID", "baseMean", "log2FC", "pval", "padj")
  # Get data from the uploaded file
  df <- read.csv(input$file$datapath, sep = ";")
  #Check data format, colnames and GeneID
  if(ncol(df)!=6)
  {
    shinyalert("Column Error","Uploaded Data has the wrong number of columns\n Need 6 columns : GeneName, ID,
    baseMean, log2FC, pval, padj",
    type="error")
    returnvalue()
  }
  else if(identical(tolower(good),tolower(colnames(df)))==FALSE)
  {
    shinyalert("Column names Error",paste0("wrong format for column ",which(colnames(df) != good)," , it must be : ",
    good[which(colnames(df) != good)],"\n",collapse=""),type = "error")
    returnvalue()
  }
  else {
    enable("start")
  }
})
```

Vérification du format du fichier

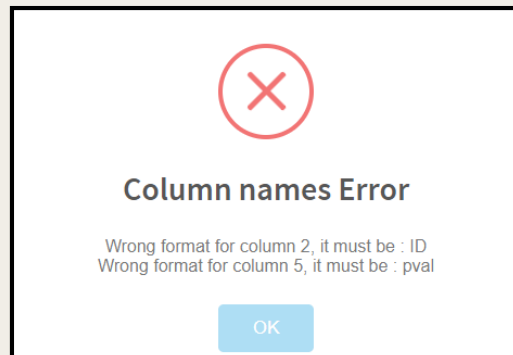
→ Noms des colonnes

→ csv



Vérification du format des données

→ EnsemblID uniquement



```
observeEvent(input$start, {
  req(input$file)
  # Get data from the uploaded file
  data_prev <- reactive({
    req(input$file)
    df <- read.csv(input$file$datapath, sep = ";")
    df["log2padj"] <- -log2(df["padj"])

    if(startswith(df$ID[1], "ENS")){
      df
    }
    else{
      shinyalert("wrong ID format !","Need Ensembl ID", type="error")
      return(NULL)
    }
  })
})
```


```
gsego <- reactive({  
  req(gene_list(), organism(), input$go_ontology, input$go_pvalue)  
  return(get_gsego(gene_list(), organism(), input$go_ontology, input$go_pvalue))  
})
```

```
output$table_go_gsea <- DT::renderDataTable({  
  req(gsego())  
  df <- as.data.frame(gsego())  
  df <- df[, -which(names(df) %in% c("core_enrichment"))]  
  Links <- paste0('<a href="https://www.ebi.ac.uk/QuickGO/GTerm?id=', df$ID, '">GO link</a>')  
  df <- df %>% add_column(Links, .after = "ID")  
  return(df)  
},
```

```
get_gsego <- function(gene_list, organism, process, pvalue){  
  gsego <- gseGO(geneList=gene_list,  
    ont =process,  
    keyType = "ENSEMBL",  
    nPerm = 10000,  
    minGSSize = 3,  
    maxGSSize = 800,  
    pvaluecutoff = pvalue,  
    verbose = TRUE,  
    orgDb = organism,  
    pAdjustMethod = "BH")  
  return(gsego)  
}
```

Liens cliquables pour GO

- Utilisation d'une fonction codée à la main pour récupérer le tableau
- Appel de cette fonction pour générer le tableau de résultat
- Utilisation de ce tableau pour tous les plots ORA



ID	Links	Description
mmu05168	KEGG link	Herpes simplex virus 1 infection
mmu04140	KEGG link	Autophagy - animal
mmu05165	KEGG link	Human papillomavirus infection
mmu04150	KEGG link	mTOR signaling pathway

```

get_table_ORA_domains <- function(data, organism, pvalue_lim){

  # Name vector with ENTREZ ids
  names(domain_gene_list) <- data3$Y

  # Omit any NA values
  domain_gene_list <- na.omit(domain_gene_list)

  # Sort the list in decreasing order (required for clusterProfiler)
  domain_gene_list = sort(domain_gene_list, decreasing = TRUE)

  # DOMAINS ANNOTATION FOR INTEREST LIST
  dataset = org_to_ensemldb(organism)
  ensembl = useMart(biomart = "ensembl", dataset = dataset)

  refseqids = names(domain_gene_list)
  domains = getBM(attributes = c("entrezgene_id", "refseq_mrna", "interpro", "interpro_description"),
                  filters = "refseq_mrna",
                  values = refseqids,
                  mart = ensembl)

  # REFERENCE GENE LIST
  gene_ref <- keys(org.Mm.eg.db, "ENTREZID")

  # DOMAINS ANNOTATION FOR REFERENCE LIST
  domain_ref_id <- keys(org.Mm.eg.db, "REFSEQ")
  domain_ref = getBM(attributes = c("entrezgene_id", "refseq_mrna", "interpro", "interpro_description"),
                    filters = "refseq_mrna",
                    values = domain_ref_id,
                    mart = ensembl)
  gene_ref <- keys(get(organism), "ENTREZID")

  # DATATABLE : Interpro ID, m, X, BgRatio, GeneRatio, pvalue, adjusted pvalue
  table <- data.frame(interproID = unique(domain_ref$interpro), domain = unique(domain_ref$interpro_description))
  k = length(gene_list) #total nb of genes in the interest list
  n = length(gene_ref) #total nb of genes in the reference list
  table$m <- table(domain_ref$interpro)[table$interpro] # nb of annotated genes in the reference list
  table$x <- table(domains$interpro)[table$interpro] # nb of annotated genes in the interest list
  table <- na.omit(table) # remove genes that are in the reference list but not in the interest one
  table$BgRatio <- signif(100*table$m/(table$m+n), 3) # compute background ratio for each domain
  table$GeneRatio <- signif(100*table$x/k, 3) # compute gene ratio for each domain
  table$pvalue <- signif(phyper(table$x-1, table$m, n, k, lower.tail = FALSE), digits = 6) # compute p-value for each domain
  table$padjust <- signif(p.adjust(table$pvalue, method = "hochberg"), digits = 6) # compute adjusted p-value for each domain

  # FINAL RESULTS
  res <- table[c("interproID", "pvalue", "padjust", "BgRatio", "GeneRatio", "X", "domain")]
  res.signif <- res[res$padjust <= pvalue_lim,]
  res.signif <- res.signif[order(res.signif$padjust),]

  return(res.signif)
}

```

```

domains_ORA_results <- reactive({
  req(data_domain(), organism())
  return(get_table_ORA_domains(data_domain(), organism(), input$domain_pvalue))
})

```

Analyse ORA

- Utilisation d'une fonction codée à la main pour récupérer le tableau
- Appel de cette fonction pour tous les plots ORA


```
box2 <- function(...){
  box(
    status = "danger",
    solidHeader = TRUE,
    width = 12,
    ...
  )
}
```

```
box2(
  title = "Gene ontology settings",
  selectInput("go_ontology",
    label = NULL,
    choices = c("Biological process" = "BP", "Molecular function" = "MF",
               "Cellular component" = "CC", "All" = "ALL")
  )
)
```

Création simplifiée de box

```
download <- function(title, plotname){
  downloadHandler(
    filename = function() { title },
    content = function(file) {
      ggsave(file, plot = plotname, device = "png")
    }
  )
}
```

```
output$download_go_barplot <- download(
  paste('barplotORA.png', sep=''),
  GO_ORA_barplot_input()
)
output$download_go_dotplot <- download(
  paste('dotplotORA.png', sep=''),
  GO_ORA_goplot_input()
)
```

Création simplifiée de boutons de téléchargement

```
output$MA <- renderPlotly2({
  req(MAPlot)
  p <- MAPlot()
  as_widget(p) %>%
    onRender(addHoverBehavior) %>%
    config(modeBarButtons = list(list("zoomIn2d"), list("zoomOut2d"), list("select2d"), list("resetScale2d"), list("toImage")))
})
```

```
renderPlotly2 <- function (expr, env = parent.frame(), quoted = FALSE){
  if (!quoted) {
    expr <- substitute(expr)
  }
  shinyRenderwidget(expr, plotlyOutput, env, quoted = TRUE)
}
```

Création simplifiée de plots

Difficultés

Conflits merge avec GitHub

Nouvelle version de R sur la fin (avril 2022)

Problème avec la sélection des organismes



Contournements

Bonne organisation + merge à la main

Garder l'ancienne version pour l'application

Se limiter aux organismes annotés seulement



Points positifs et négatifs



- Travail en équipe qui permet de développer ses compétences en apprenant les uns des autres
- Confronter les différentes idées du groupes et trouver la méthode la plus optimale pour chaque partie du projet
- Assez de créneaux de travail perso pour pouvoir travailler ensemble

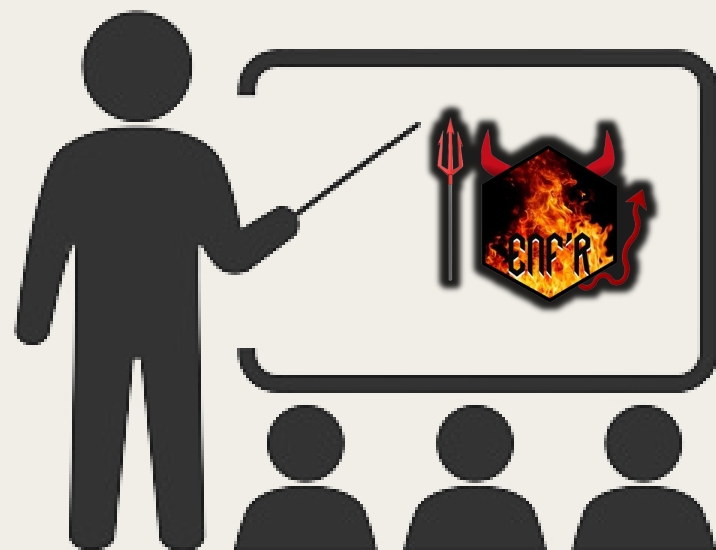
- Beaucoup (beaucoup) d'erreurs à gérer car multiples packages
- Dépendant de la surcharge des serveurs, de la connexion internet
- Manque de cours sur Shiny
- Pas de cahier des charges suffisamment précis



- ➔ Passer de GitHub à GitLab pour améliorer le travail collaboratif
- ➔ Déployer l'application sur un serveur web pour faciliter l'utilisation aux biologistes
- ➔ Accepter d'autres ID que EnsemblID (ENTREZID, ...)
- ➔ Améliorer les performances, la gestion des erreurs, la robustesse des analyses
- ➔ Intégrer d'autres analyses en fonction des besoins
- ➔ Trouver de meilleurs jeux de données

Merci de votre attention !





Présentation de l'application