```
create.fw.list <- function(db, folder = NULL, type = NULL, ecosyst=FALSE,
ref=FALSE, spatial=FALSE, code=FALSE)
  {

  #### Arguments  ####
  #'db' - database - eb (EcoBase), gw (GlobalWeb), wl (Web of Life) and mg
(Mangal)
  #'folder' - folder in the WD to get the dataset files (db=gw and wl).
  #'type' - if db=mg the user should provide the type of interactions to be
downloaded
  # 'ecosyst' - Getting ecosystem information, only for gw, eb
  #'ref' references information
  #'spatial' - get spatial info, only for wl, eb and mg

  #### Data Sources ####
  #Global Web: https://www.globalwebdb.com/
  #EcoTroph Example (EcoBase): http://sirs.agrocampus-ouest.fr/EcoTroph/index.php?action=
examples
  #Script for EcoBase: http://sirs.agrocampus-ouest.fr/EcoBase/#discoverytools
  #Mangal: #https://mangal-wg.github.io/rmangal/articles/rmangal.html
  #Web of Life: http://www.web-of-life.es/

  #### Results  ####
  fwlist <- list()

  #### Conditions for data entry  ##############################
##############################

  #db

  if(!db %in% c("eb","wl","gw","mg"))stop("Argument 'db' must take one of the
following values:\n
                                          'wl' - Web of Life
                                          'mg' - mangal
                                          'gw' - globalweb
                                          'eb' - ecobase")

  #folder
  if(!db %in% c("wl","gw") & !is.null(folder)) stop("Argument 'folder'can only
be used if 'db'= 'wl' or 'gw'!")

  #type

  #folder
  if(!db %in% c("mg") & !is.null(type)) stop("Argument 'type'can only be used if
'db'= 'mg'!")

  #ecosyst

  if(!db %in% c("gw", "eb") & ecosyst==TRUE) stop("Argument 'ecosyst'can only be
used if 'db'= 'eb' or 'gw'!")

  #ref
  #all
```

```
  #spatial
  if(!db %in% c("wl", "mg", "eb") & spatial==TRUE) stop("Argument 'spatial'can
only be used if 'db'= 'eb', 'mg' 'wl'!")

  #code
  if(!db %in% c("wl", "mg", "gw") & code==TRUE) stop("Argument 'code'can only be
used if 'db'= 'wl', 'mg'', 'gw'!")

  ############################################################
################################
  #Updating each dataset database

  #GlobalWeb
  if (db == "gw"){
    message("###################### GLOBALWEB DATABASE
######################\n\n")

    message("Fetching info from the provided folder!")
    files_gw <- list.files(path = folder, pattern = "WEB")
    ngw <- length(files_gw)
    message (paste0("There are ", ngw, " food web files in the folder!"))

    #Load files into list
    #And create vector of references
    if (ref==TRUE) reflist_gw <- c()

    names_gw <- c()#FW names

    #getting the files into R
    for(i in 1:ngw){

      message(paste0("Fetching food web ", i, " in ", ngw, "!"))

      dfgw <- read.csv(paste0(folder,"/",files_gw[i]), header = FALSE)  # read
csv file
      dfgw <- dfgw[, colSums( is.na(dfgw) ) <=1]#Remove columns with all NA

      #Get the FW name
      names_gw[i] <- as.character(dfgw[2,1])

      #Get the reference to the vector
      if (ref==TRUE) reflist_gw[i] <- as.character(dfgw[1,1])

      #to name the columns
      names_gw_c <- c()
      n1 <- ncol(dfgw)-1
      for(j in 1:n1){
        names_gw_c[j] <- as.character(dfgw[2,j+1])
      }

      #to name the rows
      names_gw_r <- c()
      n2 <- nrow(dfgw)-2
      for(j in 1:n2){
        names_gw_r[j] <- as.character(dfgw[j+2, 1])
      }
```

```
      dfgw <- dfgw[-c(1,2),-1]

      #Remove columns with NA
      dfgw[dfgw==""] <- NA
      dfgw <- na.omit(dfgw)

      if(i==281){names_gw_r <- names_gw_r[-c(36,37)]}#the FW on i=281 has a note
at the bottom

      #Delete the 'empty names'
      names_gw_c <- names_gw_c[names_gw_c!=""]
      names_gw_r <- names_gw_r[names_gw_r!=""]

      #Same names in rows or columns?
      #if(length(unique(names_gw_r)) < length(names_gw_r)) rown[i] <-
as.character(i)
      #if(length(unique(names_gw_c)) < length(names_gw_c)) coln[i] <-
as.character(i)

      #For some strange reason some rows and columns have the same name
      names_gw_c <- paste0("sp_", as.character(1:length(names_gw_c)),
"_",names_gw_c)
      names_gw_r <- paste0("sp_", as.character(1:length(names_gw_r)),
"_",names_gw_r)

      colnames(dfgw) <- names_gw_c
      rownames(dfgw) <- names_gw_r

      fwlist[[i]] <- dfgw

    }

    #Name the list
    names(fwlist) <- names_gw

    if(ref==TRUE){

      references <- as.data.frame(matrix(ncol = 4))
      names(references) <- c("FW code", "first_author", "year", "full_ref" )

      files_gw <- list.files(folder, pattern = "WEB")

      message("Fetching references from the dataset files!")

      for(w in 1:ngw){

        dfgw <- read.csv(paste0(folder,"/",files_gw[w]), header = FALSE)  # read
csv file
        #message(paste0("Reading file ", files_gw[w]))
        dfgw <- dfgw[, colSums( is.na(dfgw) ) <=1]#Remove columns with all NA

        #Get the reference to the vector
        full_ref1 <- as.character(dfgw[1,1])
        references[w,4] <- full_ref1#full reference
        references[w,1] <- files_gw[w]#fw code
        references[w,2] <- str_sub(word(full_ref1, start = 1), 1,
str_length(word(full_ref1, start = 1))-1)#fisrt author
```

```
        references[w,3] <- regmatches(x = full_ref1,gregexpr("[0-9]+",text =
full_ref1))[[1]][1]#year
        #references[w,3] <- gsub('.+\\(([0-9]+)\\).+?$', '\\1', full_ref1)#year

      }#end loop to add refs

    }#end gw refs

    #ECOSYSTEM
    if(ecosyst==TRUE){
      message("Searching for 'gw_list.csv' file...")

      if (!file.exists(paste0(folder, "/gw_list.csv"))) stop("\nThe pdf
'gw_list.pdf' has to be previously converted to a csv file...")

      #I had to conver the gw_list.pdf file to excel (csv), since I could not
install tabulizes to extract pdf tables
      gw_eco <- read.csv(paste0(folder,"/","gw_list.csv"), header = TRUE, sep =
";")  # read csv file

      filn <- paste0("WEB", as.character(gw_eco[,1]), ".csv")

      gw_eco2 <- gw_eco[,1:3]
      gw_eco2[,1] <- filn

      names(gw_eco2)[1] <- "FW"

      #yes... I do know the following few lines are 'ugly'...
      filn <- as.data.frame(cbind(filn, filn))

      names(filn) <- c("filn1","filn2")

      #files_gw <- list.files(path = folder, pattern = "WEB")

      ecosystem <- merge(x=filn, y=gw_eco2, by.x= "filn2", by.y = "FW")

      ecosystem <- ecosystem[,c(2, 3, 4)]

      names(ecosystem)[1] <- "Food web"

    }

  }#end of gw

  #Web of Life
  if (db == "wl"){

    message("###################### WEB OF LIFE DATABASE
######################\n\n")

    files_wl <- list.files(path = folder, pattern = "FW")
    nwl <- length(files_wl)
    message (paste0("There are ", nwl, " food web files in the folder!"))

    #Get refs and metrics table
    if (file.exists(paste0(folder, "/references.csv"))) {
      table_wl <- read.csv(paste0(folder, "/references.csv"), header = TRUE)  #
```

```
read csv file
    } else {
      stop("There is no 'references.csv' file on the folder, as provided by the
website!")
    }

    #FW names
    names_wl <- as.character(table_wl[,8])

    #Load files
    for(i in 1:nwl){
      message(paste0("Fetching food web ", i, " in ", nwl, "!"))
      dfwl <- read.csv(paste0(folder, "/",files_wl[i]), header = TRUE)  # read
csv file
      #row.names(dfwl) <- as.character(dfwl[,1])
      #dfwl <- dfwl[,-1]
      dfwl[is.na(dfwl)] <- 0
      fwlist[[i]] <- dfwl
    }

    names(fwlist) <- names_wl

    #REFERENCES
    if(ref==TRUE){

      references <- as.data.frame(matrix(ncol = 4))
      names(references) <- c("FW code", "first_author", "year", "full_ref" )

      message("Fetching references from the 'references.csv' file!")
      message("Checking the presence of the 'references.csv' file...")
      if(!file.exists(paste0(folder, "/references.csv"))==TRUE)stop("Can't
retrieve reference details... \n File not present!")

      ref_file <- read.csv(paste0(folder, "/references.csv"), header = TRUE)  #
read csv file

      for(w in 1:nwl){

        full_ref1 <- as.character(ref_file[w,7])
        references[w,4] <- full_ref1#full reference
        references[w,1] <- as.character(ref_file[w,1])#fw code
        references[w,2] <- str_sub(word(full_ref1, start = 1), 1,
str_length(word(full_ref1, start = 1))-1)#fisrt author
        references[w,3] <- regmatches(x = full_ref1,gregexpr("[0-9]+",text =
full_ref1))[[1]][1]#year
        #references[w,3] <- gsub('.+\\(([0-9]+)\\).+?$', '\\1', full_ref1)#year

      }#end loop to add refs


    }#end wl refs

    #SPATIAL
    if(spatial==TRUE){

      message("Fetching the spatial information from the 'references.csv'
file!")
```

```
      message("Checking the presence of the 'references.csv' file...")
      if(!file.exists(paste0(folder, "/references.csv"))==TRUE)stop("Can't
retrieve spatial info... \n File not present!")

      ref_file <- read.csv(paste0(folder, "/references.csv"), header = TRUE)  #
read csv file
      spatial1 <- ref_file[,c(1,9,10)]

    }#end of spatial

  }#end of wl

  #EcoBase
  if(db == "eb"){

    message("###################### ECOBASE DATABASE
######################\n\n")

    message("Fetching info from the EcoBase website!")
    suppressWarnings({

      #To obtain the list of available models

      suppressMessages({
        h=basicTextGatherer()
        curlPerform(url = 'http://sirs.agrocampus-ouest.fr/EcoBase/php/webser/soap-
client_3.php',writefunction=h$update)
        data1 <- xmlTreeParse(h$value(),useInternalNodes=TRUE)
        liste_mod <- ldply(xmlToList(data1),data.frame)#liste_mod contains a
list and decription
      })

      #Select only those allowing dissemination
      l2 <- subset(liste_mod, model.dissemination_allow =="true")#only those of
which dissemination is allowed
      message("Sellected only those to which model dissemination is allowed!")

      #Select only those with whole food webs
      l3 <- subset(l2, model.whole_food_web =="true")#only those with the full
food web
      message("Sellected only those to which the whole food web is available!")

      #Get model names
      model.name <- as.character(l3$model.model_name)
      input_list <- list()
      id <- as.numeric(as.character(l3$model.model_number))

      #Loop to get input list
      for(i in 1:nrow(l3)){

        message(paste0("Fetching information on food web ",i, " of ", nrow(l3)))

        suppressMessages({
          h=basicTextGatherer()
          mymodel <- id[i]
          curlPerform(url = paste('http://sirs.agrocampus-ouest.fr/EcoBase/php/webser/soap-
client.php?no_model=',mymodel,sep=''),writefunction=h$update,verbose=TRUE)
```

```
    data2 <- xmlTreeParse(h$value(),useInternalNodes=TRUE)
    input1 <- xpathSApply(data2,'//group',function(x) xmlToList(x))
  })

  #need do name the columns
  names_input <- as.character(input1[1,])
  input1 <- as.data.frame(input1)
  colnames(input1) <- names_input
  input1 <- input1[-1,]
  input_list[[i]] <- input1
}#end of loop to get input list

mnames <- names(input_list)

for (i in 1:length(input_list)){

  m2 <- input_list[[i]] #get the model

  nnodes <- length(m2)
  node_names <- names(m2)

  # if (biomass == TRUE)
  #    {
  #    nodes_biomass <- as.data.frame(matrix(ncol=3, nrow=nnodes))
  #    names(nodes_biomass) <- c("id", "name", "biomass")
  #    }

  int_matrix <- as.data.frame(matrix(ncol=nnodes, nrow=nnodes))

  for(j in 1:length(m2)){

    node1 <- m2[[j]]
    node_id <- as.numeric(node1$group_seq)
    #node1_biomass <- as.numeric(node1$biomass)
    node_name <- node_names[j]

    #biomass
    #if (biomass == TRUE)
    #{
    #nodes_biomass[node_id, 1] <- node_id
    #nodes_biomass[node_id, 2] <- node_name
    #nodes_biomass[node_id, 3] <- node1_biomass
    #}

    #matrix
    colnames(int_matrix)[node_id] <- node_name
    rownames(int_matrix)[node_id] <- node_name

    diet_node1 <- node1$diet_descr
    nr_food_items <- length(diet_node1)

    for(a in 1:nr_food_items){
      item1 <- diet_node1[[a]]
      id_item1 <- as.numeric(item1$prey_seq)
      proportion_item1 <- as.numeric(item1$proportion)
      detritus_item1 <- as.numeric(item1$detritus_fate)
      #send to matrix
```

```
            int_matrix[id_item1,node_id] <- proportion_item1
          }

        }

        int_matrix[is.na(int_matrix)] <- 0#replacing NA with 0

        #if(db=="eb" && biomass == TRUE) fwlist[[i]] <-
list(biomass=nodes_biomass, trophic_relations=int_matrix)
        #if(db=="eb" && biomass == FALSE) fwlist[[i]] <- int_matrix
        fwlist[[i]] <- int_matrix

      }

      names(fwlist) <- model.name

    })#end of outer suppressWarnings

    #REFERENCES
    if(ref==TRUE){

      references <- as.data.frame(matrix(ncol = 4))
      names(references) <- c("FW code", "first_author", "year", "full_ref" )

      message("Fetching the references information!")

      for(w in 1:nrow(l3)){

        #Get the reference to the vector
        full_ref1 <- as.character(l3$model.reference)[w]
        references[w,4] <- full_ref1#full reference
        references[w,1] <- as.numeric(as.character(l3$model.model_number[w]))#fw
code
        references[w,2] <- as.character(l3$model.author[w])#fisrt author
        references[w,3] <- regmatches(x = full_ref1,gregexpr("[0-9]+",text =
full_ref1))[[1]][1]#year
        #references[w,3] <- gsub('.+\\(([0-9]+)\\).+?$', '\\1', full_ref1)#year

      }#end loop to add refs


    }#end of eb refs

    #ECOSYSTEM
    if(ecosyst==TRUE){

      ecosystem <- data.frame(l3$model.model_number, l3$model.country,
l3$model.ecosystem_type)

      names(ecosystem) <- c("Food web", "Location", "Ecosystem")

    }#end of eb ecosystem

    #SPATIAL
    if(spatial==TRUE){

      message("Fetching spatial information from the EcoBase website...")
```

```
      #Get actual polygons
      EcoBase_shape <- sf::st_read("http://sirs.agrocampus-ouest.fr/EcoBase/php/protect
/extract_kml.php")

      ebd <- EcoBase_shape$Name

      #Getting the model numbers
      nmr <- list()
      for(i in 1:length(ebd)){
        nr <- strsplit(as.character(ebd[i]), "--::")[[1]][1]
        nr <- as.numeric(str_extract_all(nr, "\\d+")[[1]])#Alternative to
Numextract
        nmr[[i]] <- nr
      }

      nmr2 <- c()#line rows for each model
      for(i in 1:length(nmr)){
        a <- nmr[[i]]
        b <- length(a)
        c1 <- rep(i,b)
        nmr2 <- c(nmr2, c1)
      }

      #In Which row in ecobase geo file is the model?
      nmr <- unlist(nmr)
      table1 <- as.data.frame(cbind(nmr2, nmr))
      colnames(table1) <- c("row_n","id")

      #In which row does model.model_number with a given Id occurs?
      lines_n <- c()
      for (i in 1:nrow(liste_mod)){
        id <- as.numeric(as.character(liste_mod$model.model_number[i]))
        lines_n[i] <- as.numeric(table1[table1$id==id,][1])
      }

      ecobase_poly2 <- list()
      for(i in 1:length(lines_n)){
        ecobase_poly2[i] <- st_geometry(EcoBase_shape)[lines_n[i]]
        #plot(st_geometry(EcoBase_shape)[lines_n[i]], border="green", add=TRUE)
      }

      #if no polygon then bounding box
      #into here ecobase_poly2
      for(i in 1:length(ecobase_poly2)){
        if(is.na(lines_n[i])){
          #create a bounding box geographic thing
          z1 <- as.numeric(Numextract(liste_mod$model.geographic_extent[[i]]))
          z2 <- c(z1[4], z1[1], z1[2], z1[1], z1[2], z1[3], z1[4], z1[3])
          x1 <- as.data.frame(matrix(z2, ncol=2, byrow=TRUE))
          x1 <- cbind(x1[2], x1[1])#had to change lat and long... I had this the
other way around...
          p1 <- Polygon(x1)
          ps1 <- Polygons(list(p1),1)
          ecobase_poly2[[i]] <- st_as_sf(SpatialPolygons(list(ps1)))
        }
        ecobase_poly2[[i]] <- ecobase_poly2[[i]]
```

```
      }

      #convert all to class sf
      for(i in 1:length(ecobase_poly2)){
        if(!any(class(ecobase_poly2[[i]])=='sf')){
          t2 <- ecobase_poly2[[i]]
          t3 <- st_cast(t2, to="POLYGON")
          ecobase_poly2[[i]] <- st_as_sf(as(st_zm(st_geometry(t3)), "Spatial"))
        }
        else message("Ok!")
      }

      #line.Id correspondence
      table2 <- as.data.frame(cbind(1:length(ecobase_poly2),as.numeric(as.
character(liste_mod$model.model_number))))
      names(table2) <- c("row","id")

      #select the corresponding polygons
      id_selected <- as.numeric(as.character(l3$model.model_number))

      #Which rows?
      rows_selected <- c()
      for(i in 1:length(id_selected)){
        rows_selected[i]  <- as.numeric(table2[table2["id"]
==id_selected[i],][1])
      }

      spatial1 <- ecobase_poly2[rows_selected]

    }#end of eb spatial

  }#end of eb

  #MANGAL
  if(db == "mg"){

    message("####################### MANGAL DATABASE
######################\n\n")

    message("Fetching datasets from the Mangal website! \n\n Types 'predation'
and 'herbivory' by default... \n but run mangal function 'avail_type' to check
available types...\n\nThis operation might take a long time!")

    ntypes <- length(type)

    net_info <- list()

    for(i in 1:ntypes){

      message(paste0("\n\nFetching information from interactions of the type
","'",type[i], "'!"))

      fwlist1 <- search_interactions(type = type[i]) %>% get_collection()

      net_info <- rbind(net_info, fwlist1)

      fwlist2 <- as.igraph(fwlist1)
```

```
      fwlist <- c(fwlist, fwlist2)

      #class(fwlist)

    }

    #Converting igraph objects to data frame
    for(i in 1:length(fwlist)){
      fw2 <- fwlist[[i]]
      #convert each igraph to a data frame
      fw3 <- as_data_frame(fw2, what = "both")
      id_name <- fw3$vertices[,1:2]

      for(j in 1:nrow(id_name)){#clean the names

        node_name <- id_name$original_name[j]

        if (grepl(":", node_name, fixed=TRUE)) {
          node_name <- tail(strsplit(node_name, ": "))[[1]]
          id_name[j,2] <- node_name[2]
        } else id_name[j,2] <- node_name


      }#end clean names

      id_edges <- fw3$edges[,1:3]
      int_matrix <- as.data.frame(matrix(ncol = nrow(id_name), nrow =
nrow(id_name)))
      colnames(int_matrix) <- id_name$original_name
      rownames(int_matrix) <- id_name$original_name

      #Fill the matrix
      for(a in 1:nrow(id_edges)){
        edge1 <- as.numeric(id_edges[a,1:2])
        name1 <- id_name[as.character(edge1[1]),][,2]
        name2 <- id_name[as.character(edge1[2]),][,2]
        int_matrix[name1,name2] <- 1
      }

      int_matrix[is.na(int_matrix)] <- 0 #convert all NA to zero

      fwlist[[i]] <- int_matrix

    }#end of loop to convert to a data frame

    if(ref==TRUE){
      references <- as.data.frame(matrix(ncol = 4))
      names(references) <- c("Dataset ID", "first_author", "year", "DOI" )

      message("Fetching references!")

      for(j in 1:length(net_info)){
      dataset_id <- net_info[[j]]$dataset$dataset_id
      first_author <- net_info[[j]]$reference$first_author
      year_mng <- as.numeric(net_info[[j]]$reference$year)
      doi_mng <- net_info[[j]]$reference$doi
```

```
      references[j,1] <- dataset_id
      references[j,2] <- first_author
      references[j,3] <- year_mng
      references[j,4] <- doi_mng

      references <- references[order(references$`Dataset ID`),]
      rownames(references) <- 1:nrow(references)
      }

    }#End of mg refs

    if(spatial==TRUE){
      spatial1 <- as.data.frame(matrix(ncol = 4))
      names(spatial1) <- c("Dataset ID", "first_author", "lat", "long")
      message("Fetching coordinates!")

      for(z in 1: length(net_info)){
        dataset_id <- net_info[[z]]$dataset$dataset_id
        lat_mng <- net_info[[z]]$network$geom_lat
        long_mng <-  net_info[[z]]$network$geom_lon
        first_author <- net_info[[z]]$reference$first_author
        if(length(unlist(lat_mng))>1){

          spatial2 <- as.data.frame(matrix(ncol = 4))
          names(spatial2) <- c("Dataset ID", "first_author", "long", "lat" )

          for(b in 1:length(unlist(lat_mng))){
            spatial2[b,3] <- long_mng[[1]] [b]
            spatial2[b,4] <- lat_mng [[1]] [b]
            }

          spatial2[,1] <- dataset_id
          spatial2[,2] <- first_author

          spatial1 <- rbind(spatial1, spatial2)

        }
        spatial1[z,1] <- dataset_id
        spatial1[z,2] <- first_author
        if(length(unlist(lat_mng))==1) spatial1[z,3] <- lat_mng
        if(length(unlist(lat_mng))==1) spatial1[z,4] <- long_mng
      }

      spatial1 <- spatial1[order(spatial1$`Dataset ID`),]
      rownames(spatial1) <- 1:nrow(spatial1)

    }#End of mg spatial

    if (exists("references") & exists("spatial1")) (if(nrow(references)!=nrow(
spatial1)) message("WARNING: There are more than on FW in some datasets!
References and Spatial data frames have different number of rows."))

  }#end of mangal

  message(paste0("DONE! \n\nOverall the list stores ", length(fwlist), "
datasets!"))
```

```
  master_list <- list()
  master_list[["int_matrix"]] <- fwlist

  if(ecosyst==TRUE) {
    master_list[["ecosystem"]] <- ecosystem
    message ("\n Additional element in the results: \n\n The vector with
information on the ecosystems.")
  }

  if(ref==TRUE) {
    master_list[["references"]] <- references
    message ("Additional element in the results! \nA data frame with information
on the references.")
  }

  if(spatial==TRUE) {
    master_list[["spatial_info"]] <- spatial1
    message ("\n Additional element in the results: \n\n Spatial information was
added.")
  }

  if(code==TRUE) {
    if(db == "gw") master_list[["code"]] <- files_gw
    if(db == "wl") master_list[["code"]] <- files_wl
    if(db == "mg") master_list[["code"]] <- references[1,]

    message ("Added food web code information.")
  }

  #Return results
  if(length(master_list)==1) return(fwlist)
  if(length(master_list)!=1) return(master_list)

message("####################### DONE! ######################")


}#END OF FUNCTION create.fw.list
```