

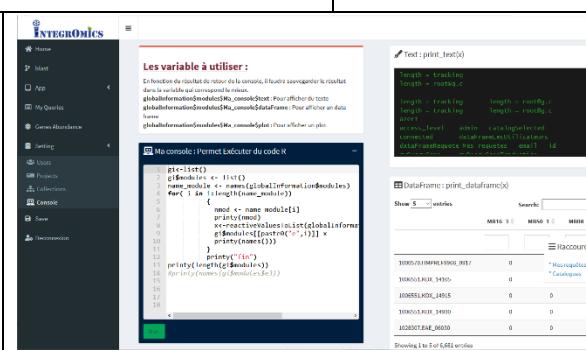
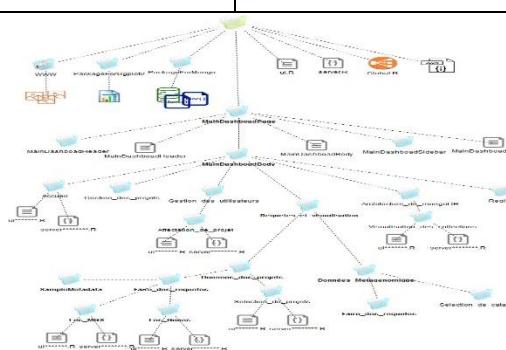
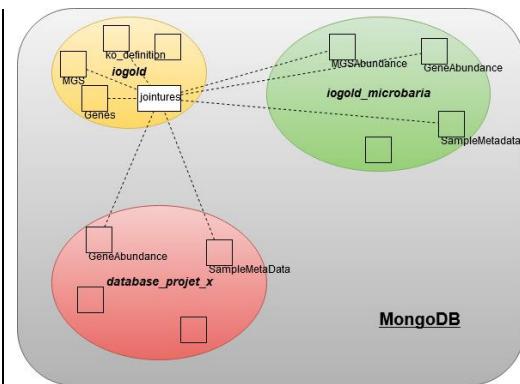
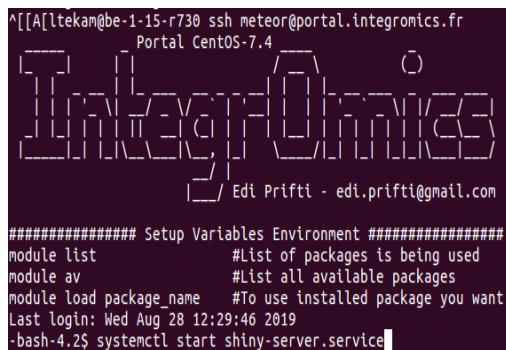
# Documentation du projet iogold

## 23 Avril – 02 Septembre 2019



ICAN, 47-83 boulevard de l'hôpital,  
75013 Paris

Développement d'une application Web permettant de faire des requêtes et des analyses sur des données métagenomiques



Ludovik TEKAM ( [tekamludovik23@gmail.com](mailto:tekamludovik23@gmail.com) )

Elève Ingénieur en 2<sup>ème</sup> Année Informatique – Sup Galilée - Université paris 13

## Responsable du projet

Eugení BELDA ( [e.belda@ican-institute.org](mailto:e.belda@ican-institute.org) )

Edi PRIFTI ( [e.prifti@ican-institute.org](mailto:e.prifti@ican-institute.org) )

4 Septembre 2019



# Sommaire

Introduction .....	5
I. Présentation du projet .....	6
1. Objectifs du projet .....	6
2. Informations.....	7
3. Connaissances requises .....	7
II. Structuration et description de l'arborescence de l'application .....	8
1. Structure global.....	8
2. Description du contenu des fichiers et dossiers .....	8
III. Configuration de l'application.....	19
1. Exemple de configuration .....	20
2. Description <i>des variables</i> .....	20
IV. Présentation de l'architecture de la base de données et description des différentes collections.....	22
1. Architecture de la base de données .....	22
2. Description des collections .....	23
V. Création et administration de la base de données.....	29
1- Prérequis .....	29
2- Ajouter un catalogue.....	29
3- Ajouter un projet.....	32
4- Ajouter une collection.....	35
5- Ajouter des données via d'autre format que le Json .....	36
VI. Méthode d'optimisation des requêtes sur la base de données .....	37
VII. Description des variables principales.....	38
1. App .....	38
2. globalInformation .....	39
VIII. Présentation des modules et perspectives d'évolution.....	40
1. Les principaux modules.....	40
2. Les modules fonctionnels .....	46
IX. Ajout de nouveaux modules .....	67
X. Astuce et conseil .....	68
1. Astuce.....	68

2. Conseil.....	68
XI. Perspective d'évolution .....	69
1. Niveau applicatif .....	69
2. Niveau structuration du code .....	69
3. Niveau interface.....	69
4. Niveau base de données .....	69
5. Niveau sauvegarde des données .....	69
6. Niveau Conception.....	70
7. Niveau fonctionnalité.....	70
XII. Mise en production de l'application .....	71
XIII. Liens utile .....	72
1. Documentations et astuce .....	72
2. Aide à la Résolution de problème.....	73
Table des matières.....	76
Annexe .....	83
iogold database population .....	83
Microbaria iogold database population.....	89

## Introduction

Ce document a pour but de présenter le projet IOGOLD qui consiste au développement d'une application web destiné au analyse bio-informatique des données méta génomique. Vous y trouverez ainsi une présentation du projet, une documentation complète sur la structuration de l'application, les modules mis en place, les idées d'amélioration ainsi que les pratiques à adopter pour bien reprendre le projet en main. Il contient également une description de la structure de la base de données et des collections qui interagissent avec l'application.

Lors de la rédaction de cette documentation, l'application web est accessible via le lien suivant « <http://iogold.integromics.fr/> ». Le code source de l'application quant à lui développement est récupérable sur le GitLab d'INTEGROMICS ( <https://git.integromics.fr/> ) projet « ioglodG ».

# I. Présentation du projet

Initialement, le titre du projet était : **Développement d'une base de données et de son interface gérant l'annotation d'espèces métagénomiques.** Cependant, étant donné qu'une première version de la base de données avait été mise en place, j'ai choisi de modifier le titre en fonction des tâches que j'ai effectué et des tâches à venir : **Développement d'une application Web permettant de faire des requêtes et des analyses sur des données méta génomiques.**

## 1. Objectifs du projet

### a. Contexte

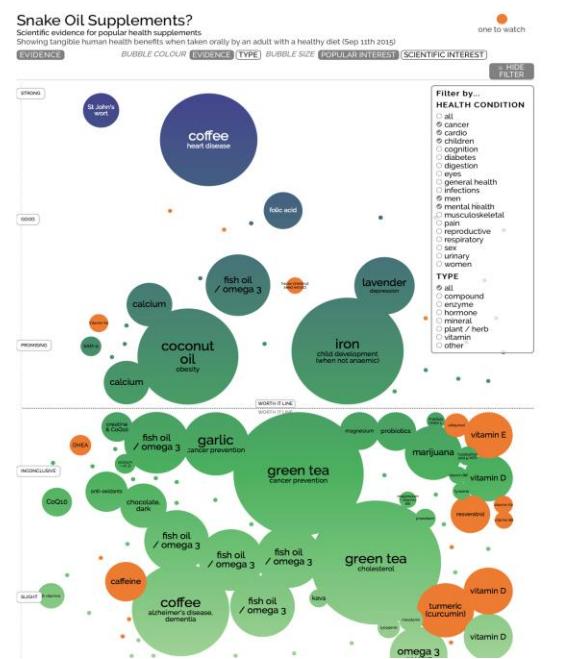
Le microbiome intestinal humain (i.e. ensemble des microorganismes habitant notre intestin) est fortement associé à notre santé. Sa dérégulation ou dysbiose peut être à l'origine ou refléter de multiples maladies chroniques humaines, comme l'obésité, le diabète, différents cancers, les maladies inflammatoires de l'intestin etc. Ces espèces bactériennes sont difficiles à étudier car il est compliqué de les isoler. La métagénomique est un nouveau domaine en forte explosion qui permet d'analyser très précisément la composition du microbiote ainsi que ses fonctions.

Différents projets internationaux ont généré de grands volumes de données de séquences génomiques (fragments d'ADN) à partir des milliers d'échantillons de l'intestin humain. Ces données ont été utilisés pour reconstruire des catalogues de gènes de référence du microbiome humain représentent le contenu génétique des espèces microbiennes le constituant. Par des approches d'apprentissage non supervisés nous avons pu établir un catalogue d'espèces métagénomique (MGS), qui sont définis comme des ensembles de gènes provenant du même génome bactérien basé sur la corrélation des abondances des gènes sur plusieurs échantillons. Certaines de ces MGS sont des cibles d'intérêt et peuvent aider à traiter différentes maladies chroniques. Le problème est qu'on connaît peu de choses sur ces entités et qu'il y en a beaucoup qui ont des interactions entre elles. Nous souhaiterions rassembler de la connaissance pertinente et construire des outils qui nous permettent de mieux voir et comprendre ces espèces et leur rôle dans la physiopathologie de l'hôte.

### b. Objectifs

Le premier objectif de ce projet est de créer une base de données relativement simple permettant de ranger des annotations (fonctionnelles et autres) qui découlent des gènes individuels les composants, mais aussi au niveau de l'espèce (implication dans une maladie donnée, etc).

Le deuxième objectif est de créer une interface de gestion et de requête de cette base mais aussi de visualisation des résultats. A titre d'exemple la figure ci-contre illustre une représentation dynamique des résultats d'une base de données sur les suppléments alimentaires<sup>1</sup>. L'exploration facile et visuelle de la base des MGS proposé dans ce projet devrait



<sup>1</sup> <http://www.informationisbeautiful.net/visualizations/snake-oil-supplements/>

permettre aux chercheurs de l'Institut du Cardiométabolisme et de la Nutrition d'accélérer leur recherche sur le rôle important des bactéries intestinales dans la prise en charge des maladies chroniques.

#### c. Existant

A l'heure actuelle nous avons commencé le développement d'un pipeline automatique d'annotation pour les gènes des catalogues méta génomiques qui intègre des informations fonctionnelles des différentes ressources (KEGG, SEED, InterPro, CARD). Ce projet permettra d'étendre ce travail et d'intégrer toutes ces informations fonctionnelles dans une seule base explorable facilement.

Il existe également un premier travail de développement de la base de données ainsi qu'une interface de requête en Java que nous mettons à disposition du stagiaire.

Plus précisément les objectifs du stage sont les suivants :

1. Prendre en main la version existante de la base de données appelé « ioGold » distribué également avec une machine virtuelle.
2. Mettre à jour ou construire de novo le schéma UML de la version existante (si satisfaisante) ou bien en créer une de novo.
3. Mise en production dans les serveurs de l'équipe (collaboration avec Ingénieur Système) de la base de données. Attention : certaines tables sont très volumineuses avec des dizaines de millions d'entrées. Il est important de faire le choix (en concertation avec les ingénieurs de l'équipe) d'une base qui peut gérer ce volume de données.
4. Peupler une nouvelle instance de la base de données avec les données du projet metacardis. Créer éventuellement de nouvelles tables pour les annotations non existantes dans la version actuelle. Tester le requettage.
5. La deuxième partie du stage concernera le développement d'une interface IHM RShiny qui permet de se connecter à la base et d'effectuer les requêtes nécessaires tout en ayant la possibilité de visualiser graphiquement certains résultats. Le stagiaire sera aidé dans cette partie.
6. Écrire un tutoriel détaillé de l'implémentation et l'utilisabilité de l'outil ainsi que le rapport de stage

## 2. Informations

**Dr. Edi Prifti et Dr. Eugeni Belda**

Institut de Cardiométabolisme et Nutrition (ICAN)

Département IntegrOmics

Maison ICAN (Bat. Claude Bernard) 1er étage.

Hôpital la Pitié Salpêtrière

50/52 bd Vincent AURIOL

75013, Paris, FRANCE

<http://www.ican-institute.org>

[e.prifti@ican-institute.org](mailto:e.prifti@ican-institute.org)

[e.belda@ican-institute.org](mailto:e.belda@ican-institute.org)

## 3. Connaissances requises

SGBD, SQL, NoSQL, web dynamique, R, Shiny

## II. Structuration et description de l'arborescence de l'application

### 1. Structure global

L'arborescence des fichiers de l'application se présente comme suit :

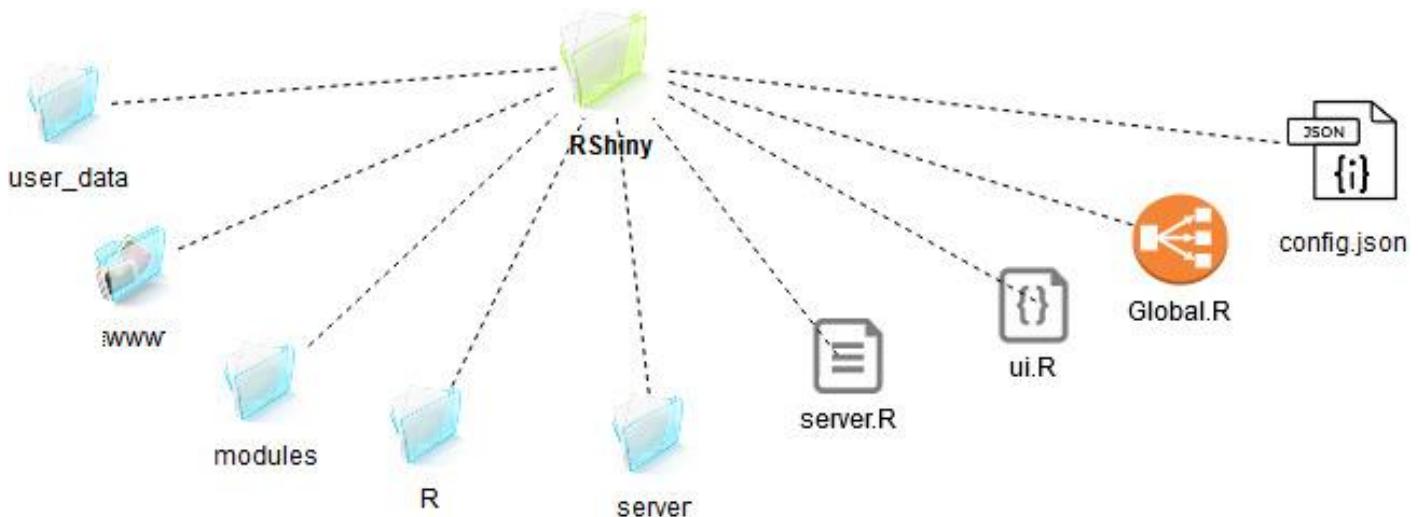


Figure II-1 : Arborescence des fichier de l'application

### 2. Description du contenu des fichiers et dossiers

#### a. Config.json

C'est le fichier de configuration de l'application. Il contient toutes les informations qui permet de lancer et de faire tourner l'application donc notamment (voir configuration de l'application pour plus de détaille)

#### b. Global.R, ui.R et server.R

Ce sont les fichiers principaux de l'application, c'est dans ces fichiers que sont charger les fonctions et variables global

##### i. Global.R

C'est le plus important des trois fichiers principaux de l'application. Il contient :

- La liste des librairies utiliser lors du développement

```
# Shiny stuff
library(shiny)
library(shinyjs)
library(shinyAce)
library(shinyCSSloaders)
library(shinydashboard)
library(shinyWidgets) # pickerInput (voir module réglage)

# data parsing
library(readr)#read_file

# database
library(mongolite)
library(jsonlite)
library(DT)
library(bcrypt) #password hashing

# vizualisation
library(ggplot2)
library(gridExtra)
library(plotly)
library(pheatmap)

# data processing
library(reshape)
library(stringr) #use for str_sub
library(dplyr) #distinct

# for module bast
library(RLinuxModules)
library(seqinr)
library(RFLPtools)
```

- Les instructions d'appel des fonctions que j'ai développé
- Une variable app, qui contient les informations sur le chargement des modules, la liste des utilisateurs connecter ... (voir description des variables principales)

#### *ii. U.i.R*

Permet de définir la structure visuelle de l'application. Dans ce projet je travaille avec shinyDashboard, c'est donc sa structure que j'utilise. (<https://rstudio.github.io/shinydashboard/structure.html#boxes>)

### iii. server.R et Server

C'est ici qu'est définie la structure des variables de chaque module (Server /serverVariable.R), les appels au serveur de chaque module (code qui permet d'exécuter les tâches en fonction de l'ui défini)

Le dossier server contient 3 fichiers qui sont l'extension du fichier serveur.R. Le fichier serveur étant assez long j'ai décidé de le diviser, mais vous pouvez, si vous le souhaitez, tout copier dans un seul fichier.

- ❖ [Le fichier serveurLesVariables.R :](#)

Il contient les variables réactives utilisées par les différents modules. Ces variables sont toutes accessibles via **globalInformation** de cette manière au lieu de passer plusieurs paramètres aux fonctions de type serveur (contenu dans chaque modules) on n'utilisera qu'une seule variable. (Exemple : `serverConnexion(input, output, session, globalInformation)` )

C'est également dans ce fichier que sont déclarés les variables de chaque module :

```
globalInformation$modules <- reactiveValues(
  # ** Catalogue ----
  Catalogue= reactiveValues(
    time = Sys.time(),
    save_session_time = 1,
    variables = list(),

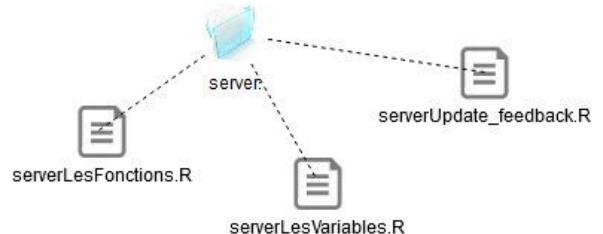
    CatalogSelected="",
    defSelected = list(),
    databaseDefSelected=list(),
    # liste des DataFrame sauvegarder par l'utilisateur
    listSaveDataFrame = list(),
    # liste des nom des DataFrame sauvegarder par l'utilisateur
    #listNameSaveDataFrame=list(),
    # le data frame qui contiendra les contrainte de la requête déjà exécuté par
    l'utilisateur
    dataFrameContrainte = data.frame(),
    dataFrameRequete = data.frame(),

    dataFrameMesRequetes = data.frame(),

    dataFrameCollectionData = list(),
    dataFrameCollectionDataValider = list(),

    dataFrameDefinition = data.frame()
  ),
  # ** Ma_console ----
  Ma_console = reactiveValues(
    time = Sys.time(),
    save_session_time = 1,
    variables = list(),

    text = "",
    dataFrame = data.frame(),
  )
)
```



*Figure II-2 : les fichiers qui contiennent l'extension du code de la partie server*

```

plot = ""
),
# ** Patient_analyses ----
Patient_analyses = reactiveValues(
  time = Sys.time(),
  save_session_time = 1,
  variables = list(),

  datajointure = data.frame()
),

```

Ces variables sont définies en fonction du besoin de chaque module. Cependant il y a des variables qui devraient être définis pour tous les modules, Il s'agit de :

- time : permet de gérer la périodicité de la sauvegarde
- save\_session\_time : nombre de minute écoulé entre deux sauvegarde
- variables = list() : est une liste qui regroupera toutes les autres variables, c'est assez pratique pour centraliser les données à sauvegarder et aussi pour les restaurer

#### ❖ Le fichier serveurLesFonctions.R :

Il contient les fonctions qui peuvent être utilisé dans le module ma console.

Pour le moment il ne contient que des fonctions d'écriture dans le log et les interfaces de sortie définie dans le modules console, mais ou pourrais rajouter d'autres fonctions en fonction des besoin d'utilisation.

### **Les variable à utiliser :**

En fonction du résultat de retour de la console, il faudra sauvegarder le résultat dans la variable qui correspond le mieux.

**globalInformation\$modules\$Ma\_console\$text** : Pour afficher du texte  
**globalInformation\$modules\$Ma\_console\$dataFrame** : Pour afficher un data frame  
**globalInformation\$modules\$Ma\_console\$plot** : Pour afficher un plot

 Ma console : Permet Exécuter du code R

```

1 printy("bonjour ", "mc")
2 printy("bonjour ", "log")
3
4
5
6
7
8

```

**Run**

 Plot : print\_plot(x)

### Text : print\_text(x)

```
bonjour
```

### DataFrame : print\_dataframe(x)

Show 5 entries      Search:

	MB16_3	MB50_1	MB0
1000570.HMPREF9966_0917	0		
1006551.KOX_14105	0		
1006551.KOX_14915	0		
1006551.KOX_14930	0	0	
1028307.EAE_06030	0	0	

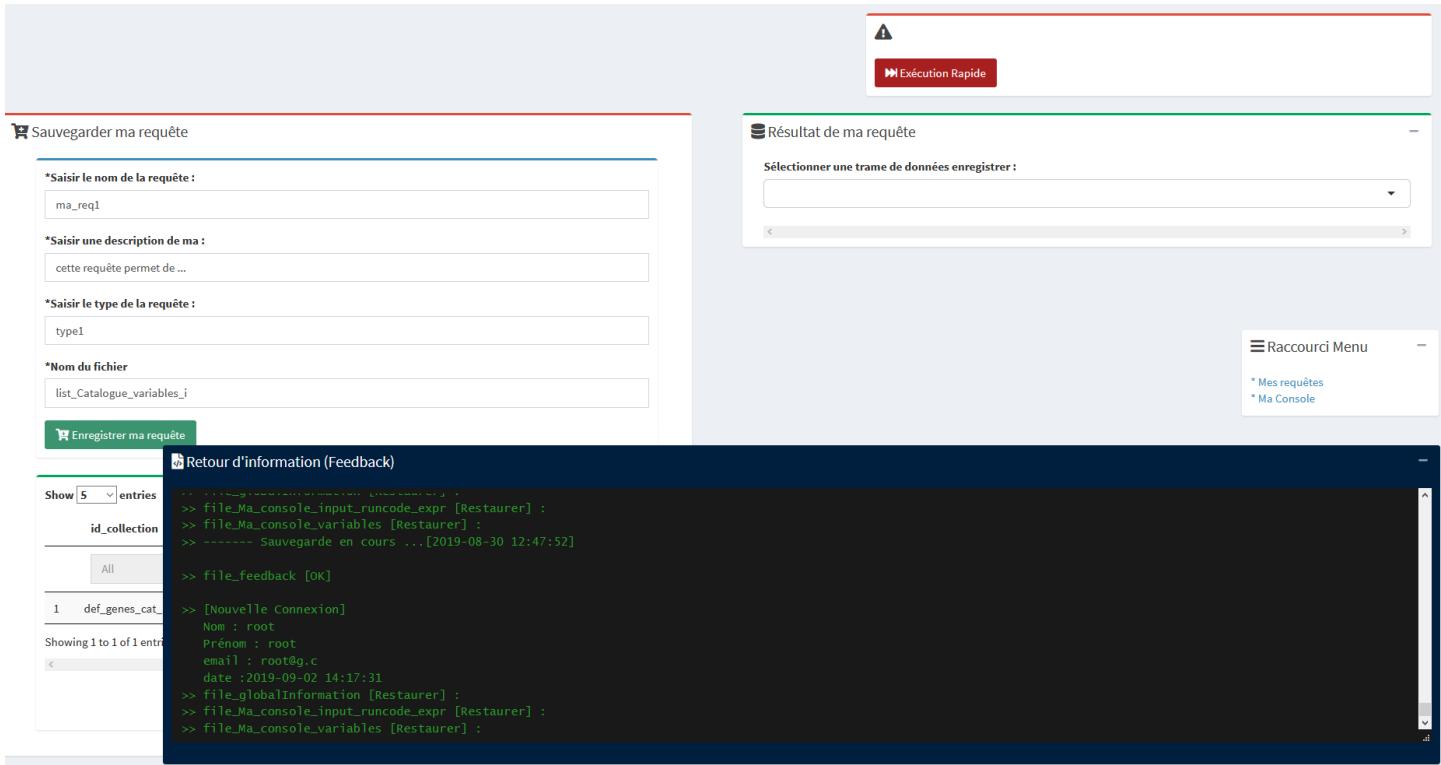
Showing 1 to 5 of 6,681 entries

Previous    **1**    2    3    4    5    ...    1337

Figure II-3 : Vue de l'onglet console

❖ Le fichier serverUpdate\_feedback.R :

Contient les lignes de code qui permettent d'actualiser les logs de chaque module. En effet chaque module a sa propre interface log.



The screenshot shows the INTEGRONICS software interface with the Catalogue module open. On the left, there is a form for saving a request, which includes fields for the request name (ma\_req1), description (cette requête permet de ...), type (type1), and file name (list\_Catalogue\_variables\_i). A green button labeled "Enregistrer ma requête" is visible. On the right, there is a log window titled "Retour d'information (Feedback)" showing command-line output. The output includes commands like "file\_Ma\_console\_input\_runcode\_expr [Restaurer]", "file\_Ma\_console\_variables [Restaurer]", and "file\_feedback [OK]". It also shows user information such as "Nom : root", "Prénom : root", "email : root@...c", and "date : 2019-09-02 14:17:31". The log window has a dark background with white text.

Figure II-4 : vu de la fenêtre de log du module catalogue

Et pour qu'elle contienne les mêmes valeurs il suffit de les mettre à jour simultanément ou presque.

```
observe({
  #cat("globalInformation")
  globalInformation$Console
  updateTextAreaInput(session ,inputId ="Console_Les_Genes" ,value =
globalInformation$Console )
  updateTextAreaInput(session ,inputId ="Console_Ma_console" ,value =
globalInformation$Console )

  updateTextAreaInput(session ,inputId ="Console_Nouveau_Module" ,value =
globalInformation$Console )
})
```

### c. Modules

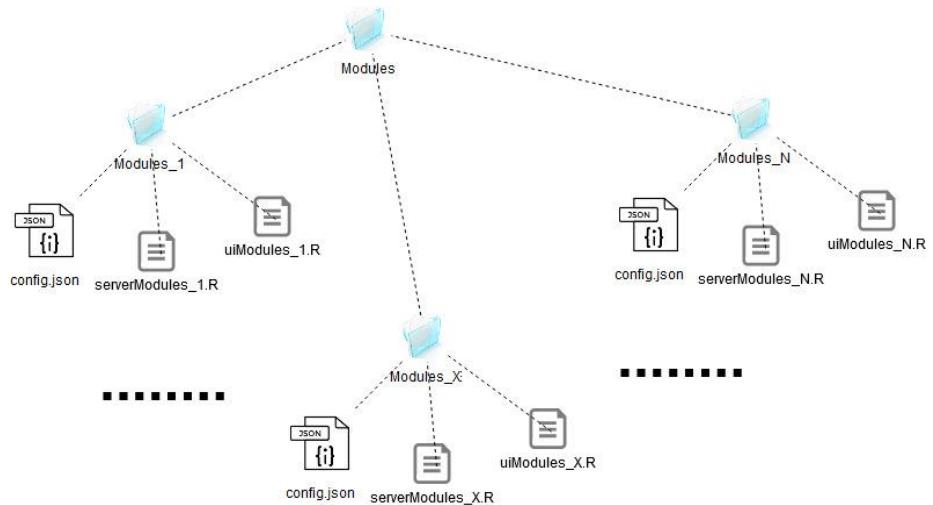
Contient les différents modules de l'application

Tous les modules ont trois fichiers communs :

- config.json
- uiModules\_X.R
- serverModule\_X.R

#### i. Config.json

Ce fichier est indispensable au chargement du module. Il contient les informations qui permet de rajouter le module créé aux modules existant. Il se présente comme suit :



```
[
  {
    "folderName": "blast",
    "ui": "tabItem_blast",
    "server": "serverblast(input, output, session, globalInformation)",
    "menu": "menuItem(text='blast', tabName = 'blast', icon = icon('code-branch'))",
    "fileToSource": ["serverblast.R", "uiblast.R"],
    "priority": 5,
    "linkModule": null,
    "rang": 10,
    "menu_en": "menuItem(text='blast', tabName = 'blast', icon = icon('code-branch'))"
  }
]
```

- **folderName** : nom du dossier qui contient le module
- **ui** : le nom de la variable qui contient la structuration de l'interface du module
- **server** : c'est la définition de la fonction dans laquelle se trouve les évènements associés au module
- **menu** : la fonction qui permet de représenter le module dans le menu ( voir <https://rstudio.github.io/shinydashboard/structure.html#sidebar-menu-items-and-tabs> )
- **fileToSource** : la liste des fichiers à charger pour que le module soit parfaitement intégré à l'application
- **priority** : c'est la priorité d'accès au module, cette variable est liée à la variable access\_level du dataframe des utilisateurs. Si le niveau d'accès d'un utilisateur est supérieur à la priorité du module, alors l'utilisateur aura accès au module, sinon il n'aura pas accès.

```
my_config_df <- app$config_df[which(globalInformation$user$access_level >= app$config_df$prior
ity & (!is.na(app$config_df$menu))),]
```

Donc un utilisateur avec un access\_level = 5 aura accès aux module de priorité inférieure ou égale à 5

- **linkModule** : la liste des modules qui sont liés à ce module, cette liste permet à l'administrateur d'avoir une idée des différentes relations entre modules
- **rang** : la position du lien du module dans le menu. Par exemple, connexion et blast sont au rang 10 alors que setting est au rang 90.
- **menu\_en** : c'est identique à **menu** mais en anglais, si on souhaite ajouter une langue il suffira de rajouter un nouveau menu et de modifier la langue via l'interface qui se trouve dans le module déconnexion

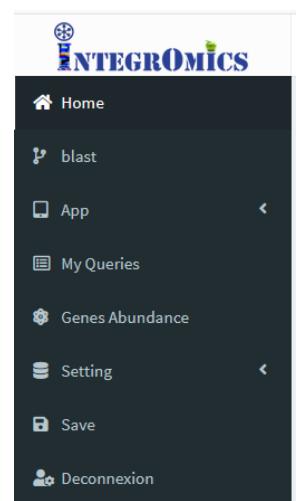
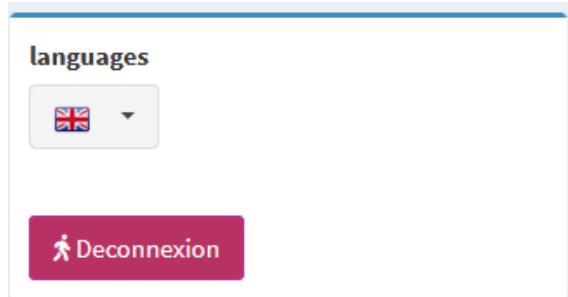


Figure II-6 : Menu administrateur

Figure II-7 : interface de déconnexion et de modification de la langue

### *ii. uiModules\_X.R*

C'est dans ce fichier qu'est défini l'interface du module, sa structure minimale est la suivante

```
tabItem_blast<-
  tabItem(
    tabName = "blast",
    fluidPage()

  ) #end tabItem
```

### *iii. serverModule\_X.R*

C'est dans ce fichier que sont définis les événements liés à l'interface créer

```
serverblast<-
  function(input, output, session, globalInformation) {

}
```

#### d. www

Contient toute les images utiliser par l'application. Exemple du logo d'integromics. On pourra aussi y place des PDF, des videos ...etc qui pourront être accessibles dans l'application

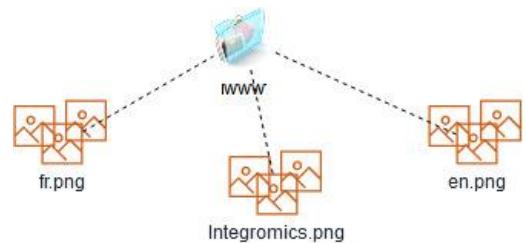


Figure II-8 : contenu du dossier www

#### e. R

Contient les différentes fonctions que j'ai définie afin de faciliter certaine

opération (la génération de plot, l'ajout de collection, la consultation de la base de données ...etc)

##### i. Arborescence

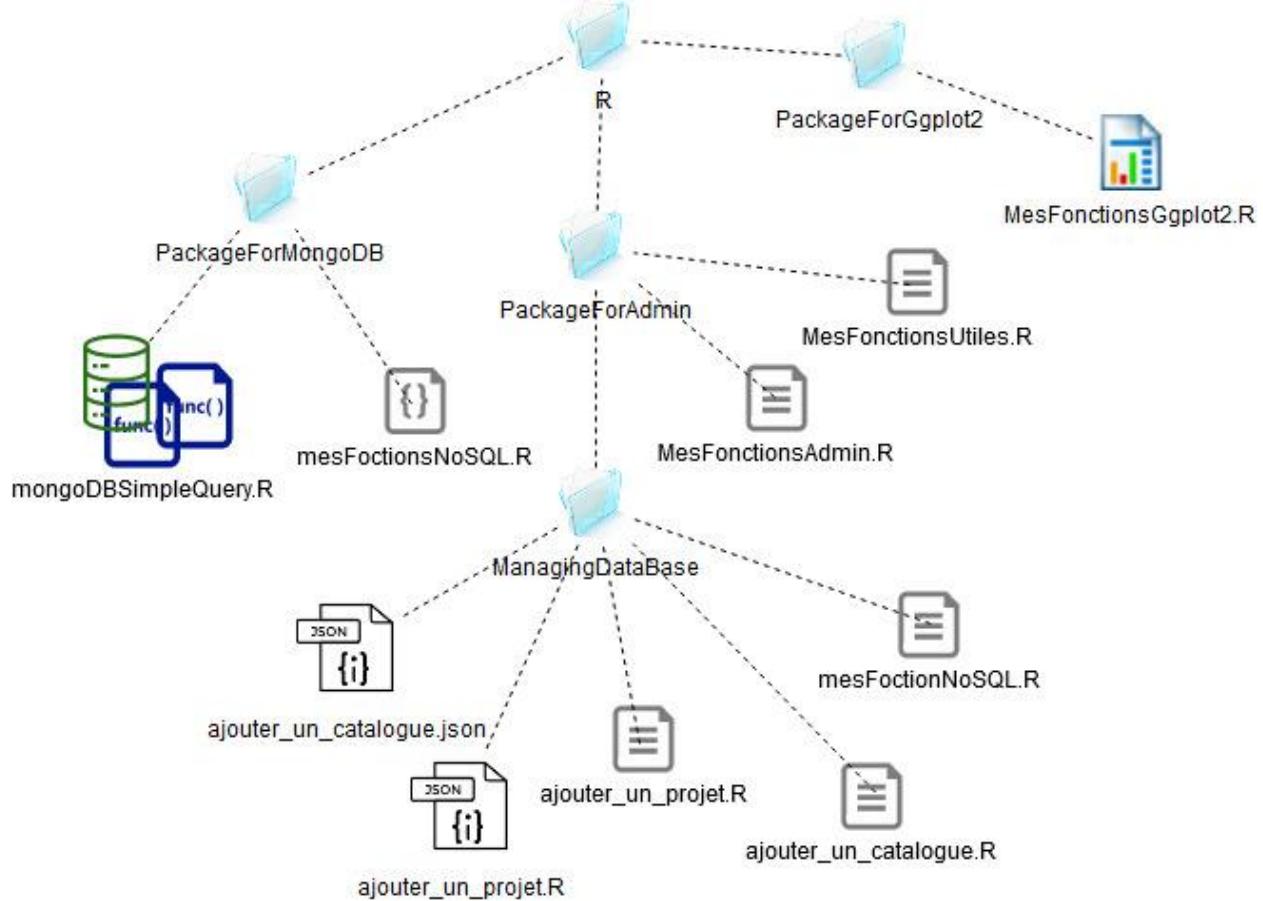


Figure II-9 : Arborescence des fichiers du dossier R

##### ii. Description

###### ❖ packageForMongoDB :

Contient les fichiers qui permettent de se connecter à la base de données et de faire des requêtes. C'est la passerelle entre l'application et la base de données.

- **mongoDBSimpleQuery.R** contient les fonctions de requête (exemple : `findQuery()`, `findAll()`, `findLimit()`, ... )

- **mesFonctionNoSQL.R** contient les fonctions qui permettent de faciliter la traduction au format json ( exemple : myQueryMinMax(), myQueryChaine(), myQueryUniqueIn(),...)

Exemple : liste des utilisateurs avec un niveau d'accès compris entre 5 et 11

```
list_user <- findQuery(databaseName = 'iogold', collectionName = 'users', myQueryJson
= myQueryMinMax(colonne = "access level", min=5, max=11))
```

Name	Password	First_name	Last_name	Admin_rights	Projects	email	profil	access_level
a	\$2a\$13\$EMa7LExfuGpWE.bCQ1uQX.bG8kVzWpW4v.b.g0q8o...	a	a	FALSE	c("testReunion", "testsoutenance")	a@g.c	chercheur	10
messi	\$2a\$16\$fgggJBjFXdf5qWKFGbAz.B42UdTyQj/OnegBle2S6VV...	leo	messi	NA	MicrobariaCopy	leo@gmail.com	administrateur	10
root	\$2a\$16\$0rwqnULJ.cnD9WtzqZ9bDeY5grMOkhL96xEdalwbS...	root	root	NA	c("MicrobariaCopy", "Microbaria", "MicrobariaCopy16")	root@g.c	administrateur	10
Prifiti	\$2a\$16\$NEm4xDDLKVRCOMFK8NBn06svtleAek2u4ocG...	Edi	Prifiti	NA	Microbaria	edi.prifiti@gmail.com	administrateur	10
belda	\$2a\$16\$cgCMGj8PZQpd.oKDIUACu/oLm34gpgzMI7XPmCi...	Eugenio	belda	NA	list("Microbaria", c("Microbaria", "MicrobariaCopy", "Micro...	e.belda@ican-institute.org	administrateur	10

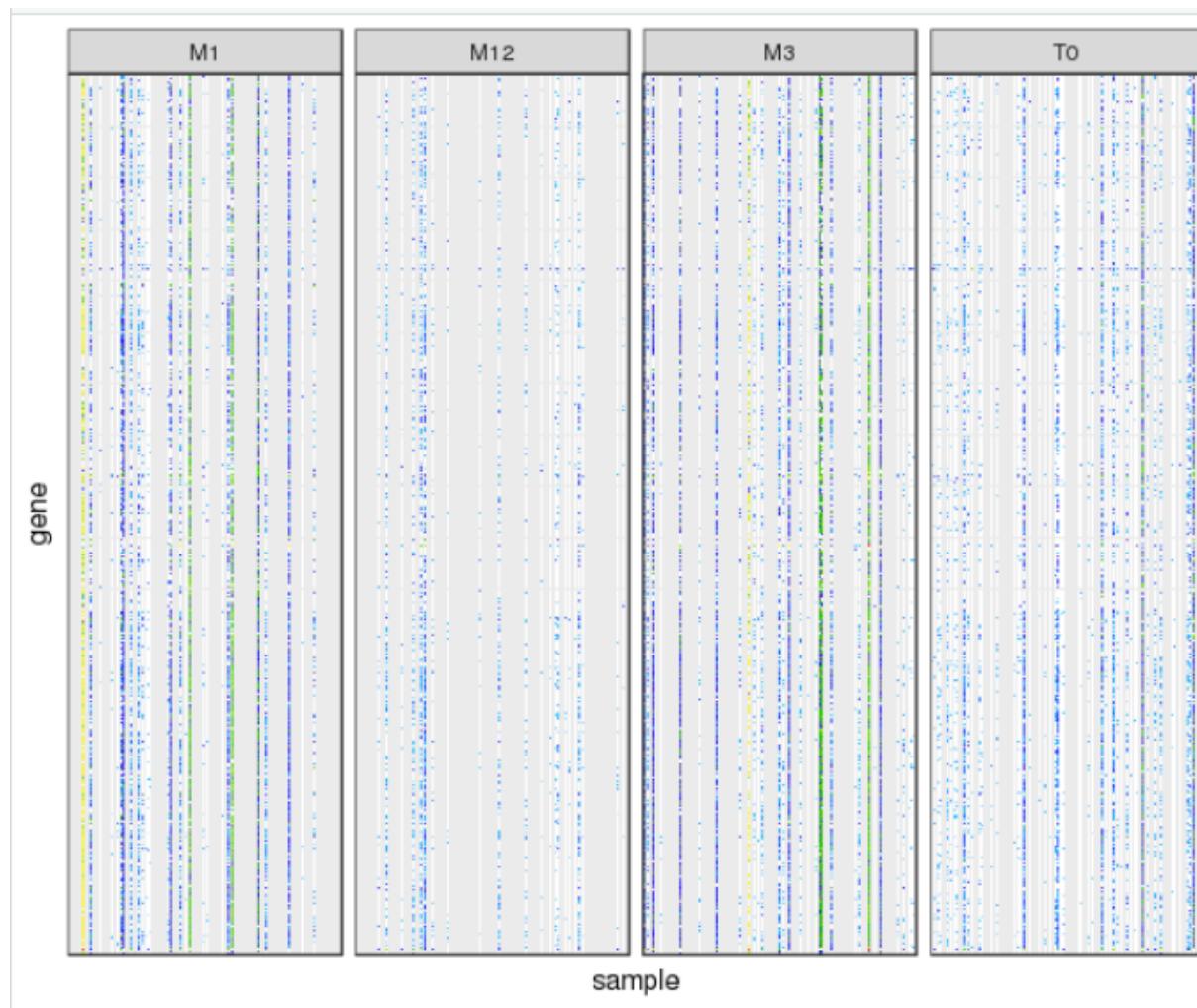
#### ❖ PackageForGgplot2 :

Contient les fonctions qui permettent de faire des plots

Exemple :

```
cag<-findAll('iogold','CAG00022')
proj<-findAll('iogold microbaria','obs samplemetadata v1 Microbaria')
monPlot<-myBarcodePlot1(cag,proj,'Time')
```

**monPlot** : bare code du MGS **GAG00022** avec une stratification par la variable **Time**



❖ **PackageForAdmin :**

- **MesFonctionUtiles.R** : Contient un ensemble de fonctions qui facilitent certaines opérations. Par exemple les fonctions d'affichages des erreurs ( uiMessage\_liste\_colonnes\_def\_catalog\_Les\_Genes(), uiMessage\_ResultatConsoleR\_Les\_Genes(),... ) , de description des catalogues (decrireCatalogue() ), et d'autres fonctions qui permettent de réduire le nombre de ligne de code de l'application.
- **MesFonctionsAdmin.R** : contient deux fonction, la fonction d'ajout de nouveau module, et la fonction d'ajout de nouvelle collection à l'aide d'un fichier json. On pourrait éventuellement rajouter de nouvelles fonctions (exemple de la fonction d'ajout de nouvelle collection au format csv, rda ... )

❖ **ManagingDataBase**

Les fonctions de ce dossier permettent d'ajouter de nouveau catalogue et projet à l'application (voir administration de la base données). Il contient également un dossier **json\_init** dans lequel j'ai mis une copie des collections que j'ai rajoutée afin de mieux gérer l'application, donc en cas de perte de données vous pourriez toujours les restaurer.

#### f. user\_data

Correspond au répertoire de gestion des sessions. Pour le moment, un répertoire plus sécurisé n'a pas encore été créé. Une fois qu'il aura été défini, il suffira de modifier le chemin dans le fichier de configuration principale (voir configuration de l'application)

##### i. Arborescence

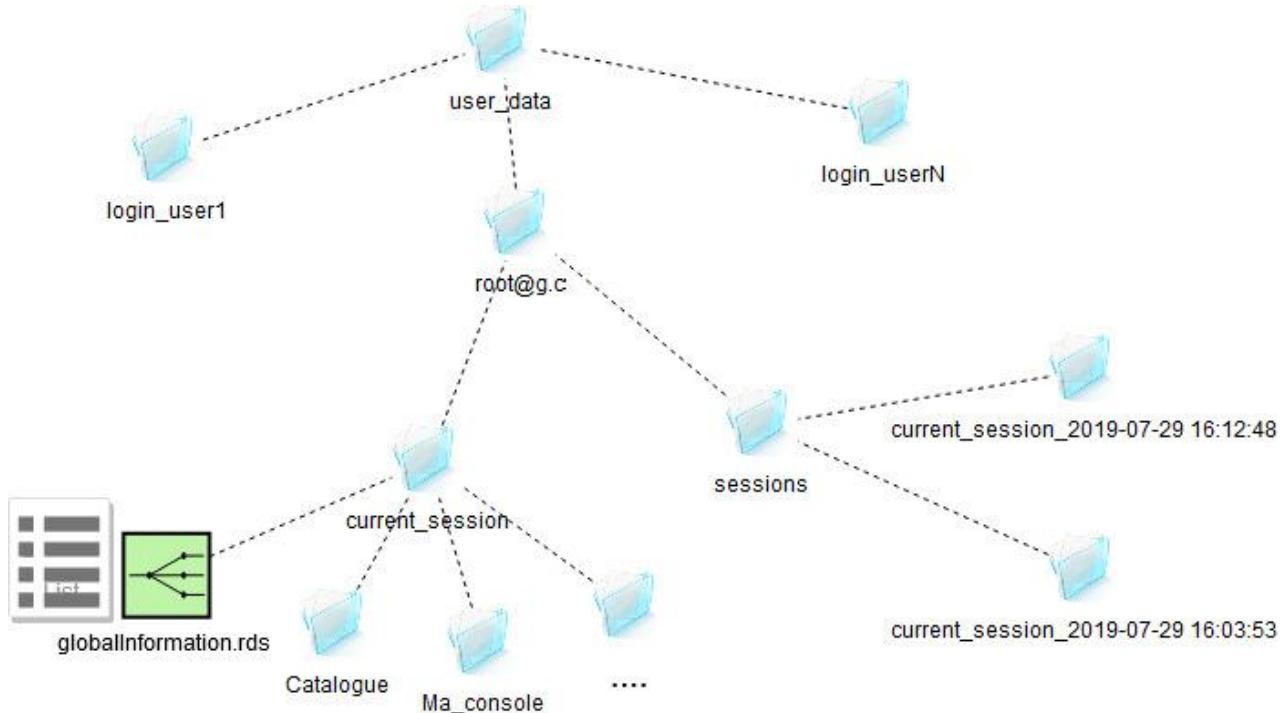


Figure II-10 : arborescence de sauvegarde des données utilisateurs

##### ii. Description

Le dossier **user\_data** contient plusieurs sous dossier, chaque sous-dossier correspond à un utilisateur.

Chaque dossier utilisateur contient deux sous-dossiers **sessions** et **current\_session**, **sessions** contient simplement les sauvegardes de **currente\_session** et **currente\_session** contient un fichier **globalInformation.rds** qui est une sauvegarde des données de cet utilisateur, il contient aussi plusieurs sous-dossier qui corresponde aux différents modules de l'application. De cette manière on pourra sauvegarde des données spécifique et propre à chaque module pour ensuite pouvoir les exploiter individuellement.

La structure de sauvegarde et l'arborescence des fichiers peut être modifier si elle ne vous convient pas. J'ai regroupé tout le code de sauvegardes et restaurations des données dans le **module sauvegarde** (voir présentation des modules : le module save/sauvegarde pour plus de détaille)

## III. Configuration de l'application

Pour configurer l'application, il suffit de modifier le fichier de configuration global ou de rajouter une nouvelle configuration.

## 1. Exemple de configuration

```
{
  "server_id" : "portal",
  "activated":0,
  "default_user_email":"root@g.c",
  "default_user_pwd":"root",
  "language" : "en",

  "workDirectory":"/share/apps/iogold/iogoldapp/user_data",
  "appDirectory":"/share/apps/iogold/iogoldapp",
  "pathToModule":"/share/apps/iogold/iogoldapp/modules/",

  "databases":["iogold","iogold_microbaria"],
  "pwd":"rT@g4CkYkaTRS=%8",
  "host" : "gpu1.integromics.fr:27017",
  "username" : "shiny",

  "globalDatabase" : "iogold",
  "usersCollection":"users",
  "usersCollectionDelete":"usersDelete",
  "projectsCollections" : "projects",
  "mainCollections" : "collections",
  "catalogCollections" : "catalog",
  "jointuresCollections" : "jointures",
  "moduleCoverageCollections":"module_coverage_KEGGdb_5_19",
  "typeColumnCollections":"typeColumn"
},
}
```

## 2. Description des variables

### Variables de sélection de la configuration à utiliser :

- server\_id : permet d'indiquer ou cette configuration doit être utiliser
- activated : permet de activer une configuration. 0 = désactiver et 1 = activer

### Variables de connection par défaut :

- default\_user\_email : email de l'utilisateur par défaut
- default\_user\_pwd : mot de passe de l'utilisateur par defaut
- language : langue utiliser au chargement de l'application

### Variables de spécification des chemins :

- workDirectory : indique le répertoire où sera sauvegarder les données des utilisateurs
- appDirectory : indique le répertoire où se situe l'application
- pathToModule : indique le répertoire où se trouve les modules de l'application

### Variables de connexion aux bases de données :

- databases : liste des base de données accessible via l'application

- pwd : mot de passe pour se connecter à ces base de données
- host : indique qui est l'hôte du SGBD. Dans notre cas MongoDB
- username : nom de l'utilisateur qui a accès à cette base de données

Variables de spécification des collections les plus fréquemment utiliser :

- globalDatabase : nom de la base de données ou se trouve les catalogue et les collection de gestion de l'application.  
Dans notre cas c'est « iogold »
- usersCollection : nom de la collection qui contient la liste des utilisateurs
- usersCollectionDelete : nom de la collection qui contient les utilisateurs supprimer
- projectsCollections : nom de la collection qui contient la liste des projets
- mainCollections : nom de la collection qui contient la liste de toute les collections
- catalogCollections : nom de la collection qui contient la liste des catalogues
- jointuresCollections : nom de la collection qui contient les liens entre les différentes collection
- moduleCoverageCollections : nom de la collection qui contient les scores des modules par rapport au MGS
- typeColumnCollections : nom de la collection qui contient le type de chaque variable de chaque collection

## IV. Présentation de l'architecture de la base de données et description des différentes collections

### 1. Architecture de la base de données

Les données sont représentées dans plusieurs bases de données qui sont requérable via l'application. Elles ne sont pas toutes les mêmes et peuvent être réparties en deux catégories. D'un côté on a **les données qualitatives** (plus connues sous le terme de **dimension** ou de définition des types de variable quantifiée), d'un autre côté nous avons **les données quantitatives** (ou les observations des différentes variables d'une dimension donnée).

Les bases de données sont constituées comme suit :

- Une base de données (actuellement « **iogold** ») qui contient :
  - Les catalogues de données qualitatives, on pourra notamment avoir plusieurs versions d'un même catalogue
  - Un ensemble de collections qui permettra de gérer les utilisateurs, les projets, les catalogues et les différents liens entre catalogue et projet.
- Un ensemble de bases de données qui représenteront les différents projets, chaque projet étant constitué des abondances (quantification des définitions d'un catalogue plus éventuellement d'autres collections comme les données des patients, les collections d'abondances de gènes pour un mgs données) relatives à un catalogue spécifique.

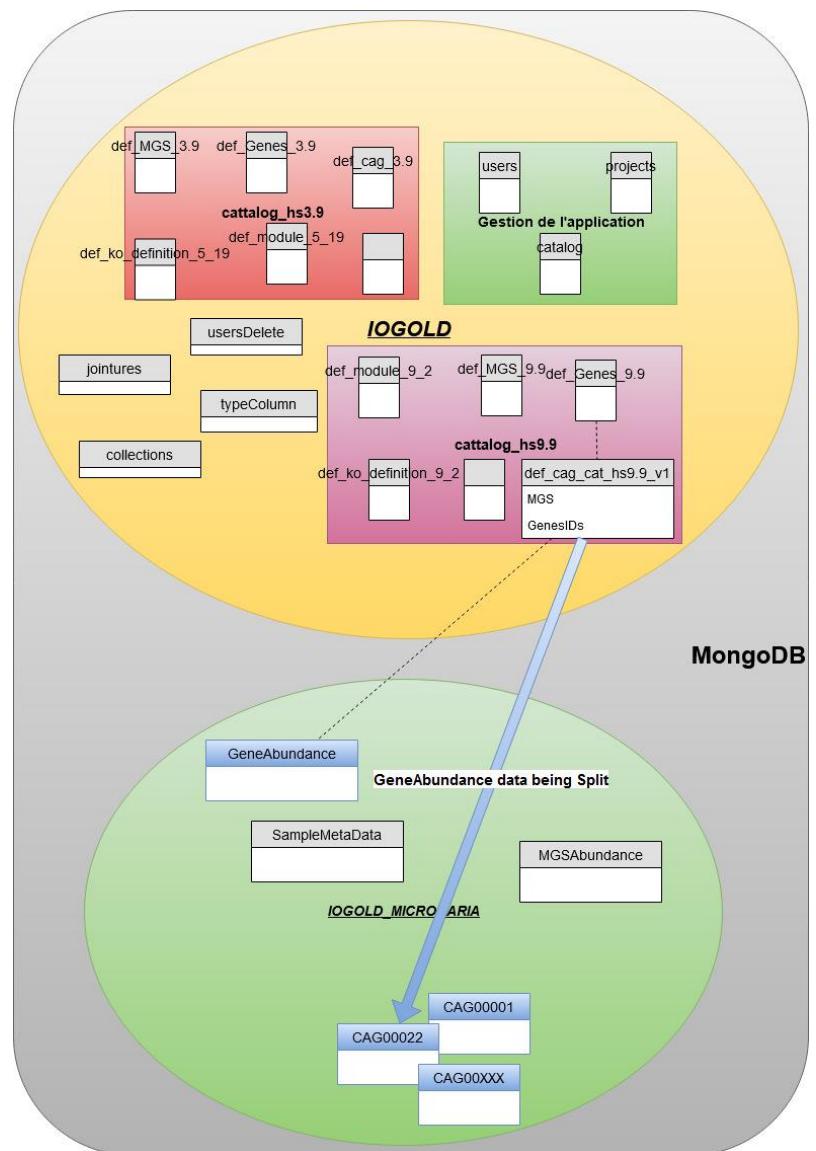


Figure III-1 : vue partielle des bases de données et collections dans MongoDB

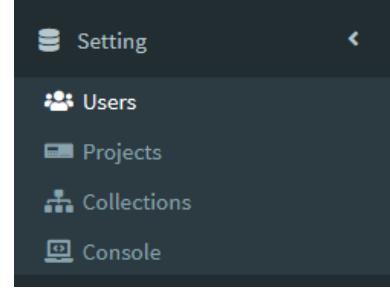
Le schéma ci-dessus est une représentation de l'architecture des bases de données dans MongoDB actuellement il n'y a que deux base de données, iogold et iogold\_microbaria.

## 2. Description des collections

Remarque : pour visualiser le contenu des différentes collections depuis l'application il vous suffit d'aller dans l'onglet **collections**. (Voir présentations du module collection pour plus de détailler)

### a. Les collections existantes

Les collections existantes sont celle qui constituent les catalogues et les projets.



#### i. Description des données

Une documentation détaillée expliquant la nature des données de ses collections est mise en annexe (*iogold database population, Microbaria iogold database population*).

#### ii. Diagramme de classe

Ci-dessous, le diagramme de classe des données des catalogues :

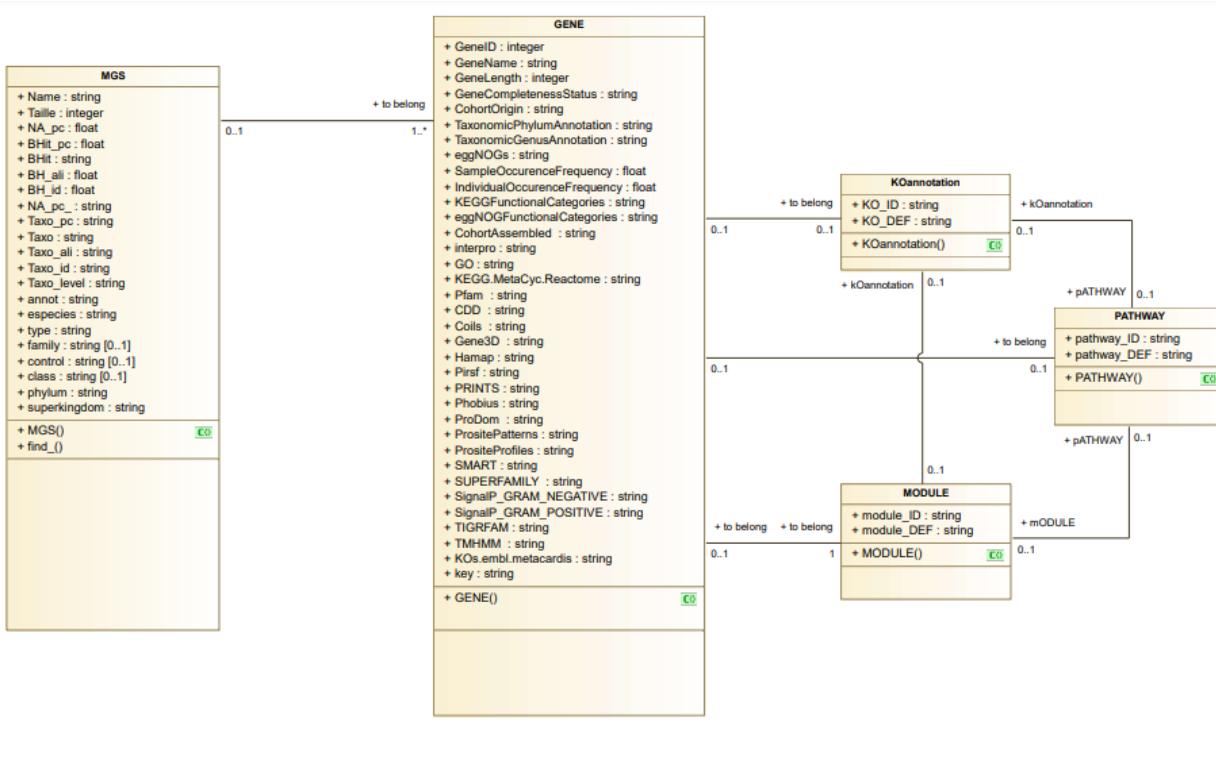


Figure IV-2 : diagramme de classe des données du catalogue

## b. Les collections rajoutées

J'ai rajouter de nouvelles collections qui permettent de gérer les utilisateurs, les projets, les catalogues et les différents liens entre catalogue et projet.

### 1. Description des données

Leur description est données ci-dessous :

#### ❖ users

Contient la liste des utilisateurs qui peuvent se connecter à l'application

- Quelques données
  - Code R

```
findLimit('iogold','users',5)
```

- Visualisation

users
email
Password
First_name
Last_name
Projects
profil
access_level
date_ajout
connecter

	email	First_name	Last_name	Projects	access_level	date_ajout
1	a@g.c	a	a	testReunion, testsoutenance	10	<NA>
2	leo@gmail.com	leo	messi	MicrobariaCopy	10	2019-06-05
3	root@g.c	root	root	MicrobariaCopy, Microbaria, MicrobariaCopy16	10	2019-06-06
4	edi.prifti@gmail.com	Edi	Prifti	Microbaria	10	2019-06-14
5	e.belda@ican-institute.org	Eugenio	belda	Microbaria, Microbaria, MicrobariaCopy, MicrobariaCopy16	10	2019-06-20

#### ❖ projects

Contient la liste des projets

- Quelques données
  - Code R

```
findLimit('iogold','projects',5)
```

- Visualisation

projects
name
collections
catalog
version
database

	name	collections	catalog	version
1	Microbaria obs_genes_cat_hs9.9_v1_Microbaria, obs_genes_mgs_cat_hs9.9_v1_Microbaria, obs_mgs_cat_hs9.9_v1_Microbaria, obs_samplemetadata_v1_Microbaria cat_hs9.9_v1	v1	v1	v1
2	MicrobariaCopy obs_genes_cat_hs9.9_v1_MicrobariaCopy, obs_mgs_cat_hs9.9_v1_MicrobariaCopy, obs_samplemetadata_v1_MicrobariaCopy cat_hs9.9_v1	v1	v1	v1
3	MicrobariaCopy16 obs_genes_cat_hs9.9_v1_MicrobariaCopy16, obs_mgs_cat_hs9.9_v1_MicrobariaCopy16, obs_samplemetadata_v1_MicrobariaCopy16 cat_hs9.9_v1	v1	v1	v1
4		NULL	v1	v1
5	Microbaria_test database	NULL	cat_hs9.9_v1	v1
1	iogold_microbaria			
2	iogold			
3	iogold			
4	iogold_microbaria			
5	iogold_microbaria			

### ❖ catalog

Contient la liste des catalogues

- Quelques données

- Code R

```
findLimit('iogold','catalog',5)
```

- Visualisation

```
> findLimit('iogold','catalog',5)
      name
1   cat_hs9.9_v1
2   cat_hs3.9_v1
3 cat_hs9.9_test_v5

collections
1 def_genes_cat_hs9.9_v1, def_genomesgs_cat_hs9.9_v1, def_mgstanatomy_cat_hs9.9_v1, def_ko_definition_KEGGdb_5_19, def_module_definitions_KEGGdb_5_19, def_ko_Module_associations_KEGGdb_5_19, def_
module_pathway_associations_KEGGdb_5_19, def_ko_pathway_associations_KEGGdb_5_19, def_cag_cat_hs9.9_v1
2   def_genes_cat_hs3.9_v1, def_mgstanatomy_cat_hs3.9_v1, def_ko_definition_KEGGdb_5_19, def_module_definitions_KEGGdb_5_19, def_ko_Module_associations_KEGGdb_5_19, def_
module_pathway_associations_KEGGdb_5_19, def_ko_pathway_associations_KEGGdb_5_19, def_cag_cat_hs3.9_v1
3
def_ko_definition_KEGGdb_30_08, def_mgstanatomy_cat_hs9.9_test_v5
  database
1 iogold
2 iogold
3 iogold

description
1 <b>cat_hs9.9_v1</b><br>Vous avez sélectionné le catalogue métagénomique <b>hs9.9_v1</b> de microbaria.<br>Il est constitué des définitions suivantes : <br> <ul><li>def_genes_cat_hs9.9_v1 (987 9896 éléments)</li> <li>def_genomesgs_cat_hs9.9_v1 (2691498 éléments)</li> <li>def_mgstanatomy_cat_hs9.9_v1 (3481 éléments)</li> <li>def_cag_cat_hs9.9_v1 (3463 éléments)</li> </ul> Il est également associer au description KEGG suivantes : <ul><li>def_ko_definition_KEGGdb_5_19 (22078 éléments)</li> <li>def_ko_Module_associations_KEGGdb_5_19 (5103 éléments)</li> <li>def_pathway_definitions_KEGGdb_5_19 (533 éléments)</li> <li>def_ko_pathway_associations_KEGGdb_5_19 (960 éléments)</li> <li>def_module_definitions_KEGGdb_5_19 (815 éléments)</li> <li>def_module_pathway_associations_KEGGdb_5_19 (960 éléments)</li></ul>
2 <b>cat_hs3.9_v1</b><br>Vous avez sélectionné le catalogue métagénomique <b>hs3.9_v1</b> de microbaria.<br>Il est constitué des définitions suivantes : <br> <ul><li>def_genes_cat_hs3.9_v1 (3871657 éléments)</li> <li>def_mgstanatomy_cat_hs3.9_v1 (2181 éléments)</li> <li>def_cag_cat_hs3.9_v1 (7381 éléments)</li> </ul> Il est également associer au description KEGG suivantes : <ul><li>def_ko_definition_KEGGdb_5_19 (22078 éléments)</li> <li>def_ko_Module_associations_KEGGdb_5_19 (5103 éléments)</li> <li>def_pathway_definitions_KEGGdb_5_19 (533 éléments)</li> <li>def_ko_pathway_associations_KEGGdb_5_19 (960 éléments)</li> <li>def_module_definitions_KEGGdb_5_19 (815 éléments)</li> <li>def_module_pathway_associations_KEGGdb_5_19 (960 éléments)</li></ul>
3
<NA>
```

catalog
name
collections
database
description

### ❖ Jointures

Contient les clés qui permettent de faire les liens entre les différentes collections des différentes bases de données

jointures
nameCollectionA
nameCollectionB
key

- Quelques données

- Code R

```
findLimit('iogold','jointures',5)
```

- Visualisation

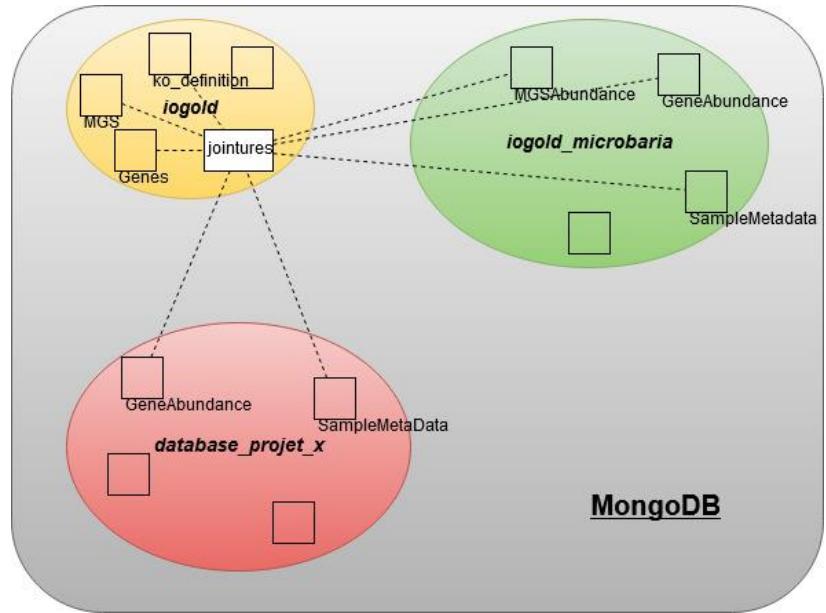


Figure IV-3 : visualisation du rôle de la collection jointure dans MongoDB

```
> findLimit('iogold','jointures',5)
      nameCollectionA           nameCollectionB      key          catalog
1   def_genes_cat_hs9.9_v1    def_genes_cat_hs9.9_v1 GeneID cat_hs9.9_v1, cat_hs9.9_v2
2   def_mgstaxonomy_cat_hs9.9_v1 def_mgstaxonomy_cat_hs9.9_v1 _row             cat_hs9.9_v1
3   def_ko_definition_KEGGdb_5_19 def_ko_definition_KEGGdb_5_19 koIDs cat_hs9.9_v1, cat_hs9.9_v2
4   def_module_definitions_KEGGdb_5_19 def_module_definitions_KEGGdb_5_19 _row cat_hs9.9_v1, cat_hs9.9_v2
5 def_ko_Module_associations_KEGGdb_5_19 def_ko_Module_associations_KEGGdb_5_19 moduleIDs cat_hs9.9_v1
```

#### ❖ usersDelete

Contient la liste des utilisateurs supprimer, cela permet de garder une trace de toute les personnes qui ont eu à utiliser cette application

- Quelques données

- Code R

```
findLimit('iogold','usersDelete',5)
```

usersDelete
email
Password
First_name
Last_name
Projects
profil
access_level
date_ajout
date_supression

- Visualisation

```
> findLimit('iogold','usersDelete',5)
      Name First_name Last_name          Password      email      profil date_ajout date_supression
1     t         t         t            t            t        root@g.c administrateur 2019-06-05 2019-06-05
2   root       root $2a$13$K8T77N5N0EJXhH011NeAB.RQsc8uXtjTJ2i5QDuIqRx4BD9j9Y3.i      NA        2019-06-06
3 ebeldauer eugeni belda $2a$13$t571nE35BQucv6HX5D6guxpoRGryDQuVWf6li5ydKPSfitL6ZUSi      NA        NA        2019-06-06
4   Goat       Goat Goat $2a$13$UxWp2vg7UARGFYIRZ079duxv5mohni3g5s1wmvqNfIbrXN5g3nlia      NA        NA        2019-06-06
```

#### ❖ typeColumn

Contient le type de données de toute les variables des différentes collections de catalogue et de projet. Très utile au niveau applicatif afin de déclarer les variables avec le bon type de données. Cette collection est très utilisée dans le module qui permet de générer des requêtes. Si le type de données des variables d'une collection n'ai pas défini dans cette collection, elle ne pourra pas être requêteable (voir les fonction **typeColumnCollectionDefinition** et **typeColumnCollectionObservation** qui sont défini dans le fichier **mongoDbSimpleQuery** et qui permet de mettre à jour cette collection lors de l'ajour de nouvelle collection dans la base de données)

typeColumn
collection
column
type
min
max

- Quelques données

- Code R

```
findLimit('iogold','typeColumn',5)
```

- Visualisation

```
> findLimit('iogold','typeColumn',5)
      collection          column      type min      max
1 def_genes_cat_hs9.9_v1      GeneID integer  1 9879896
2 def_genes_cat_hs9.9_v1      GeneName character  0  0
3 def_genes_cat_hs9.9_v1     GeneLength integer 100 88230
4 def_genes_cat_hs9.9_v1 GeneCompletenessStatus character  0  0
5 def_genes_cat_hs9.9_v1     CohortOrigin character  0  0
```

### ❖ Collections

Contient la liste des collections utiliser par l'application. A l'exception des collection CAGXXXXX qui sont assez nombreuse et n'ont pas vraiment besoin d'être listé.

- Quelques données

- Code R

```
findLimit('iogold','collections',5)
```

- Visualisation

collections
name
number_lines
number_columns
size
type
database
catalog
description
version
project

```
> findLimit('iogold','collections',5)
      name number_lines number_columns size   type database      catalog
1 collections          19            11    1 iogold  iogold cat_hs9.9_v1
2 projects              1             1    1 iogold  iogold cat_hs9.9_v1
3 users                 16            7    1 iogold  iogold cat_hs9.9_v1
4 def_genes_cat_hs9.9_v1 9879896        36    1   def  iogold cat_hs9.9_v1
5 def_genemsgs_cat_hs9.9_v1 2691408        36    1   def  iogold cat_hs9.9_v1
                                         description version      project
1 Contient les infomations des collections liées à cette application, (à l'exception des collection CAGXXXXX ) <NA>      <NA>
2 Contient la liste des projets requétables via l'application <NA>      <NA>
3 Contient la liste des utilisateurs utilisant cette application <NA>      <NA>
4 <NA>           v1 CatalogAnnotation_hs9.9
5 <NA>           v1 CatalogAnnotation_hs9.9
```

## 2. Diagramme de classe

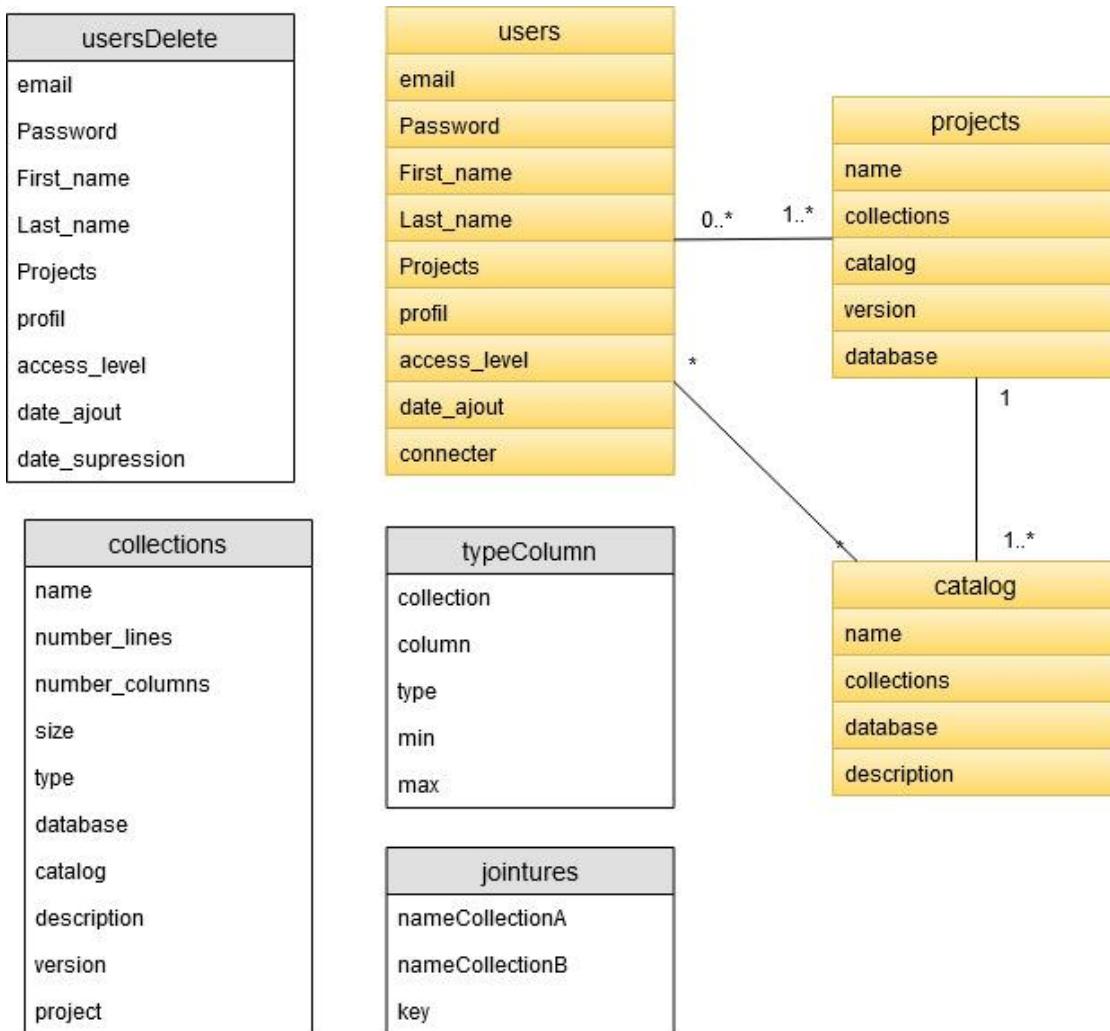


Figure IV-4 : Diagramme de classe de la gestion des utilisateurs , projets et catalogues de l'application

## V. Création et administration de la base de données

### 1-Prérequis

Les prérequis consistent à définir certaines variables et à charger en mémoire un ensemble de fonction. La méthode la plus simple pour le faire est de faire un run du fichier global.R ( ..... /RShiny/global.R) et ensuite de faire stop.

#### Étape 1

Vérifier que le fichier config.json de l'application est bien configurer en fonction de votre environnement de travail ( voir la partie sur la configuration de l'application )

#### Étape 2

- En ligne de commande dans la console :
 

```
runApp ("~/home/ltekam/GIT_LAB_test/iogoldg/RShiny")
```
- À l'aide du bouton Run App :
 
  
Ouvrez le fichier global.R et cliquer sur run app

#### Étape 3



Maintenant toutes les variables et fonctions donc nous avons besoin ont bien été chargé en mémoire.

Pour le vérifier, regarder si les fonctions suivantes sont dans votre environnement :

- ajouter\_un\_projet
- ajouter\_un\_catalogue

### 2-Ajouter un catalogue

#### Étape 1

Modifier le fichier **ajouter\_un\_catalogue.json**

- Ouvrez le dossier ..... /RShiny/R/PackageForAdmin/ManagingDataBase
- Ouvrez le fichier ajouter\_un\_catalogue.json et compléter le en respectant le type de la collection à ajouter.

Chaque ligne correspond à une collection du catalogue

❖ Les collections de type KEGG doivent respecter la forme suivante :

```
{
  "database" : "iogold",
  "catalog" : "",
  "version" : "",
  "date" : "6_14",

  "def_collection" : "def_ko_definition_KEGGdb_",
  "common_json_file_name" : "",
  "path" : ""

}
```

❖ Les collections de définition de Gène et MGS doivent respecter le format suivant :

```
{
  "database" : "iogold",
  "catalog" : "cat_hs9.9_",
  "version" : "v2",
  "date" : "",

  "def_collection" : "def_mgstaxonomy_",
  "common_json_file_name" : "",
  "path" : ""

}
```

Remarque :

- Pour les collections KEGG, le champ « catalog » et « version » doivent être vide
- Pour les collections de définition de Gene et MGS, le champ « date » doit être vide

Description des champs

- database : base de données où sera stocké le catalogue
- catalog : nom du catalogue
- version : version du catalogue
- date : date de création de la collection
- def\_collection : nom de la collection à rajouter et respectant le format suivant : def\_nomCollection\_
- common\_json\_file\_name : nom du fichier json dans lequel les données du projet sont sauvegardées.

Remarque : si les données sont stockées dans plusieurs fichiers mais avec un nom commun

Exemple : ICGgenes\_1\_1000.json / ICGgenes\_1000\_2000.json ... , dans ce cas on indique uniquement le nom commun : ICGgenes

- path : chemin vers le fichier de la collection

Remarque : le chemin doit se terminer par un /

Example: /data/db/iogold/microbiome/ 0.json.tables.ICGgenesMGS.MongoDB/

Étape 2

Exécuter la fonction :

```
ajouter_un_catalogue(JsonFileNameCollectionprojet ='ajouter_catalogue.json' ,path
='./R/PackageForAdmin/ManagingDataBase/',
catalogProjectName="CatalogAnnotation_hs9.9_2")
```

- **JsonFileNameCollectionCatalog**='ajouter\_un\_catalogue.json', qui correspond au nom du fichier qui contient les collections du catalogue
- **path**='./R/PackageForAdmin/ManagingDataBase/', le chemin vers le dit fichier **JsonFileNameCollectionCatalog**
- **catalogProjectName**="CatalogAnnotation\_hs9.9\_2", le nom du projet dans lequel s'inscrit le catalogue qui sera rajouter
- 

➤ *Exemple d'ajout de catalogue : le catalogue hs9.9*

Note :

- Par mesure de sécurité j'ai changé quelque paramètre. Si vous souhaiter réellement régénérer le projet Microbaria Il faudra :
  - Remplacer « cat\_hs9.9\_test\_ » par « cat\_hs9.9\_ »
  - Remplace « CAG5 » par « CAG »
  - Compléter les collections manquantes
- Seule les collections précises dans le fichier json seront régénérer

*Étape 1 : Je modifie le fichier ajouter\_catalogue\_hs9.9.json*

```
[
  {
    "database": "iogold",
    "catalog" : "",
    "version": "",
    "date": "30_08",

    "def_collection": "def_ko_definition_KEGGdb",
    "common_json_file_name": "kodefinitions",
    "path": "/data/db/iogold/microbiome/0.json.tables.IGCgenesMGS.MongoDB/"

  },
  {
    "database": "iogold",
    "catalog" : "cat_hs9.9_test_",
    "version": "v5",
    "date": "",

    "def_collection": "def_mgstaxonomy",
    "common_json_file_name": "CAG5",
    "path": "/data/db/iogold/microbiome/0.json.tables.IGCgenesMGS.MongoDB/"

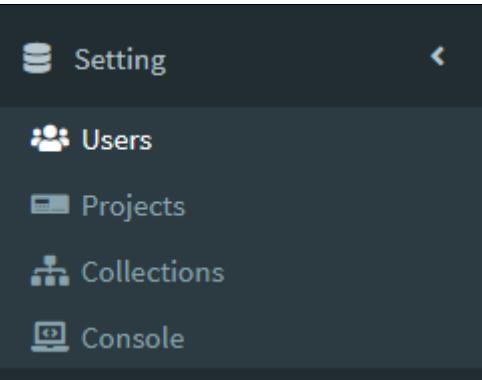
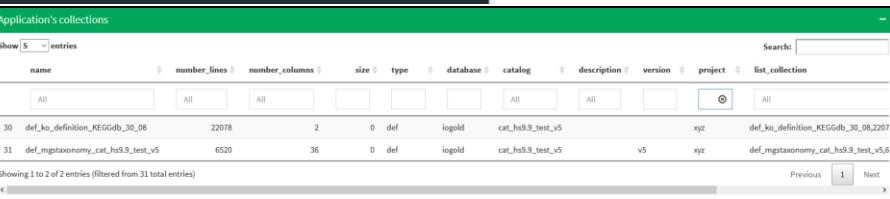
  }
]
```

### Étape 2 : J'exécute la fonction ajouter\_un\_projet :

```
ajouter_un_catalogue(JsonFileNameCollectionprojet
='ajouter_catalogue_hs9.9.json' ,path
='./R/PackageForAdmin/ManagingDataBase/',
catalogProjectName="CatalogAnnotation_hs9.9_2")
```

### Vérification

Vérifier le nouveau catalogue est accessible via l'application

Sur le menu de l'administrateur, je sélectionne collections	
j'ai une vue sur les collection du catalogue rajouter	

## 3-Ajouter un projet

### Étape 1

Demander à l'ingénieur système de rajouter une base de données pour le nouveau projet tout en respectant la configuration des base de données existante « iogold » et « iogold\_microbaria » :

```
"pwd": "rT@g4CkYkaTRS=%8",
"host" : "gpul.integromics.fr:27017",
"username" : "shiny",
```

### Étape 2

- Ouvrez le dossier ..... /RShiny/R/PackageForAdmin/ManagingDataBase
- Ouvrez le fichier ajouter\_un\_projet.json et compléter le en respectant le type de collection à ajouter.

Chaque ligne correspond à une collection du projet

- ❖ Les collection d'abondances doivent respecter la forme suivante :

```
{  
  "database" : "database of project",
```

```

"project" : "Microbaria",

"globaldatabase" : "iogold",
"catalog" : "cat_hs9.9_",
"version" : "v1",

"obs_collection" : "obs_mgs_",
"common_json_file_name" : "",
"path" : ""

},

```

❖ Les collections de métadonnées doivent respecter la forme ci-dessous :

```

{
  "database": "database_of_project",
  "project": "Microbaria",

  "globaldatabase" : "iogold",
  "catalog" : "",
  "version" : "v1",

  "obs_collection" : "obs_samplemetadata",
  "common_json_file_name" : "",
  "path" : ""

}

```

Remarque : Pour les collections métadonnées, le champ « catalog » doit être vide

#### Description des champs

- database : base de données où sera stocké le projet , celle rajouter par l'ingénieur système.
- project : nom du projet
- globaldatabase : base de données où se trouve les catalogue et les collection de gestion de l'application. Sauf déplacement de données se sera « iogold »
- catalog : nom du catalogue duquel le projet est issue
- version : version du catalogue
- obs\_collection : nom de la collection à rajouter et respectant le format suivant : obs\_nomCollection\_
- common\_json\_file\_name : nom du fichier json dans lequel les données projet sont sauvegardé.

Remarque : si les données sont stockées dans plusieurs fichier mais avec un nom commun

Exemple : MicrobariaGeneAbundance\_1000001:2000000.json /

MicrobariaGeneAbundance\_2000001:3000000.json ... , dans ce cas on indique uniquement le nom commun :

MicrobariaGeneAbundance

- path : chemin vers le fichier de la collection

Remarque : le chemin doit se terminer par un /

Example: /data/db/iogold/microbiome/2.json.tables.Microbaria/

#### Étape 3

Exécuter la fonction :

```
ajouter_un_projet(JsonFileNameCollectionprojet ='ajouter_projet.json' ,path
=' ./R/PackageForAdmin/ManagingDataBase/')
```

*Exemple d'ajout de projet : le projet Microbaria*

Note :

- Par mesure de sécurité j'ai changé quelque paramètre. Si vous souhaiter réellement régénérer le projet Microbaria Il faudra :
  - Remplacer « Microbaria\_test » par « Microbaria »
  - Remplace « MicrobariaGeneAbundance\_1000001:2000000 » par « MicrobariaGeneAbundance »
- Seule les collections précises dans le fichier json seront régénérer

*Étape 1 : Je modifie le fichier ajouter\_un\_projet.json*

```
[
  {
    "database": "iogold_microbaria",
    "project": "Microbaria_test",

    "globaldatabase" : "iogold",
    "catalog" :"",
    "version": "v1",

    "obs_collection": "obs_samplemetadata_",
    "common_json_file_name": "MicrobariaSampleMetadata",
    "path": "/data/db/iogold/microbiome/2.json.tables.Microbaria/"

  },
  {
    "database": "iogold_microbaria",
    "project": "Microbaria_test",

    "globaldatabase" : "iogold",
    "catalog" : "cat_hs9.9_",
    "version": "v1",

    "obs_collection": "obs_mgs_",
    "common_json_file_name": "MicrobariaMGSabundance",
    "path": "/data/db/iogold/microbiome/2.json.tables.Microbaria/"

  },
  {
    "database": "iogold_microbaria",
    "project": "Microbaria_test",

    "globaldatabase" : "iogold",
    "catalog" : "cat_hs9.9_",
    "version": "v1",

    "obs_collection": "obs_genes_",
    "common_json_file_name": "MicrobariaGeneAbundance_1000001:2000000",
    "path": "/data/db/iogold/microbiome/2.json.tables.Microbaria/GeneAbundance_With_row/"
  }
]
```

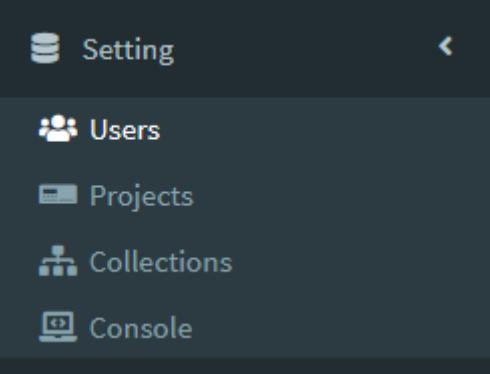
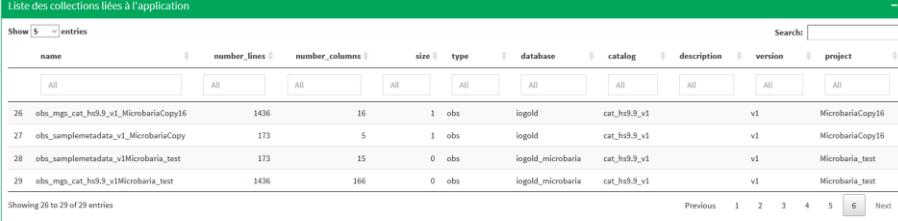
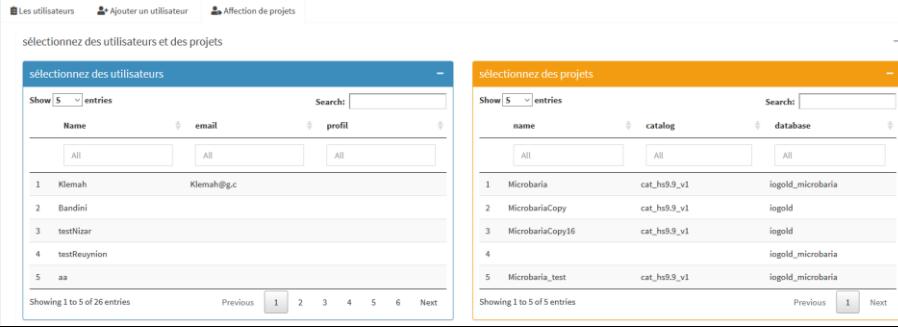
]

### Étape 2 : J'exécute la fonction ajouter\_un\_projet :

```
ajouter_un_projet(JsonFileNameCollectionprojet
='ajouter_projet_microbaria.json' ,path
='./R/PackageForAdmin/ManagingDataBase/')
```

### Verification

Vérifier le nouveau projet est accessible via l'application

<p>Sur le menu de l'administrateur, je sélectionne collections ou users</p>																																																													
<p>Si je sélectionne collections, j'ai une vue sur les collection du projet rajouter</p>	 <p>Liste des collections liées à l'application</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>name</th> <th>number_lines</th> <th>number_columns</th> <th>size</th> <th>type</th> <th>database</th> <th>catalog</th> <th>description</th> <th>version</th> <th>project</th> </tr> </thead> <tbody> <tr><td>All</td><td>All</td><td>All</td><td>All</td><td>All</td><td>All</td><td>All</td><td>All</td><td>All</td><td>All</td></tr> <tr><td>26 obs_mga_cat_hs9.9_v1_MicrobariaCopy16</td><td>1436</td><td>16</td><td>1</td><td>obs</td><td>iogold</td><td>cat_hs9.9_v1</td><td>v1</td><td></td><td>MicrobariaCopy16</td></tr> <tr><td>27 obs_samplemetadatav1_MicrobariaCopy</td><td>173</td><td>5</td><td>1</td><td>obs</td><td>iogold</td><td>cat_hs9.9_v1</td><td>v1</td><td></td><td>MicrobariaCopy16</td></tr> <tr><td>28 obs_samplemetadatav1Microbaria_test</td><td>173</td><td>15</td><td>0</td><td>obs</td><td>iogold_microbaria</td><td>cat_hs9.9_v1</td><td>v1</td><td></td><td>Microbaria_test</td></tr> <tr><td>29 obs_mga_cat_hs9.9_v1Microbaria_test</td><td>1436</td><td>166</td><td>0</td><td>obs</td><td>iogold_microbaria</td><td>cat_hs9.9_v1</td><td>v1</td><td></td><td>Microbaria_test</td></tr> </tbody> </table> <p>Showing 26 to 29 of 29 entries</p>	name	number_lines	number_columns	size	type	database	catalog	description	version	project	All	All	All	All	All	All	All	All	All	All	26 obs_mga_cat_hs9.9_v1_MicrobariaCopy16	1436	16	1	obs	iogold	cat_hs9.9_v1	v1		MicrobariaCopy16	27 obs_samplemetadatav1_MicrobariaCopy	173	5	1	obs	iogold	cat_hs9.9_v1	v1		MicrobariaCopy16	28 obs_samplemetadatav1Microbaria_test	173	15	0	obs	iogold_microbaria	cat_hs9.9_v1	v1		Microbaria_test	29 obs_mga_cat_hs9.9_v1Microbaria_test	1436	166	0	obs	iogold_microbaria	cat_hs9.9_v1	v1		Microbaria_test
name	number_lines	number_columns	size	type	database	catalog	description	version	project																																																				
All	All	All	All	All	All	All	All	All	All																																																				
26 obs_mga_cat_hs9.9_v1_MicrobariaCopy16	1436	16	1	obs	iogold	cat_hs9.9_v1	v1		MicrobariaCopy16																																																				
27 obs_samplemetadatav1_MicrobariaCopy	173	5	1	obs	iogold	cat_hs9.9_v1	v1		MicrobariaCopy16																																																				
28 obs_samplemetadatav1Microbaria_test	173	15	0	obs	iogold_microbaria	cat_hs9.9_v1	v1		Microbaria_test																																																				
29 obs_mga_cat_hs9.9_v1Microbaria_test	1436	166	0	obs	iogold_microbaria	cat_hs9.9_v1	v1		Microbaria_test																																																				
<p>Si je sélectionne user, dans l'onglet affecter projet, j'ai bien mon nouveau projet qui apparait, dans notre cas c'est Microbaria_test</p>	 <p>sélectionnez des utilisateurs</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Name</th> <th>email</th> <th>profil</th> </tr> </thead> <tbody> <tr><td>All</td><td>All</td><td>All</td></tr> <tr><td>1 Klemah</td><td>Klemah@g.c</td><td></td></tr> <tr><td>2 Bandini</td><td></td><td></td></tr> <tr><td>3 testNizar</td><td></td><td></td></tr> <tr><td>4 testReymon</td><td></td><td></td></tr> <tr><td>5 aa</td><td></td><td></td></tr> </tbody> </table> <p>Showing 1 to 5 of 26 entries</p> <p>sélectionnez des projets</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>name</th> <th>catalog</th> <th>database</th> </tr> </thead> <tbody> <tr><td>All</td><td>All</td><td>All</td></tr> <tr><td>1 Microbaria</td><td>cat_hs9.9_v1</td><td>iogold_microbaria</td></tr> <tr><td>2 MicrobariaCopy</td><td>cat_hs9.9_v1</td><td>iogold</td></tr> <tr><td>3 MicrobariaCopy16</td><td>cat_hs9.9_v1</td><td>iogold</td></tr> <tr><td>4</td><td></td><td>iogold_microbaria</td></tr> <tr><td>5 Microbaria_test</td><td>cat_hs9.9_v1</td><td>iogold_microbaria</td></tr> </tbody> </table> <p>Showing 1 to 5 of 5 entries</p>	Name	email	profil	All	All	All	1 Klemah	Klemah@g.c		2 Bandini			3 testNizar			4 testReymon			5 aa			name	catalog	database	All	All	All	1 Microbaria	cat_hs9.9_v1	iogold_microbaria	2 MicrobariaCopy	cat_hs9.9_v1	iogold	3 MicrobariaCopy16	cat_hs9.9_v1	iogold	4		iogold_microbaria	5 Microbaria_test	cat_hs9.9_v1	iogold_microbaria																		
Name	email	profil																																																											
All	All	All																																																											
1 Klemah	Klemah@g.c																																																												
2 Bandini																																																													
3 testNizar																																																													
4 testReymon																																																													
5 aa																																																													
name	catalog	database																																																											
All	All	All																																																											
1 Microbaria	cat_hs9.9_v1	iogold_microbaria																																																											
2 MicrobariaCopy	cat_hs9.9_v1	iogold																																																											
3 MicrobariaCopy16	cat_hs9.9_v1	iogold																																																											
4		iogold_microbaria																																																											
5 Microbaria_test	cat_hs9.9_v1	iogold_microbaria																																																											

## 4-Ajouter une collection

Pour ajouter une collection, il suffit d'utiliser la fonction **GenerateMyDataMultiAdmin()**

Exemple : ajout de la collections jointures, la collection qui permet de faire le lien entre les différentes collections

```
GenerateMyDataMultiAdmin(databaseName = 'iogold','jointures',jsonFileName =
"jointures",pathJsonFileName =
'/home/xxxx/GIT_LAB_test/iogoldg/RShiny/R/PackageForAdmin/ManagingDataBase/json_data_ini
t/')
```

## 5-Ajouter des données via d'autre format que le Json

Pour rajouter des données qui ne sont pas au format json, il faudra soit :

- Les convertires au format json a l'aide des fonctions déjà existante :
  - Conversion data frame vers json : exemple de la conversion de la liste des utilisateurs

```
x = findAll('iogold','users')

write_json(x ,path =
'/home/ltekam/GIT_LAB_test/iogoldg/RShiny/R/PackageForAdmin/ManagingDataBase
/json_data_init/users.json')
```

- Écrire de nouvelles fonctions de génération de données. Pour cela vous n'êtes pas obligé de partir de zero, vous pouvez vous inspire de la fonction **GenerateMyDataMultiAdmin()** et y modifier quelque valeur principalement cette ligne :

```
db$insert(fromJSON(cheminCollection))
```

## VI. Méthode d'optimisation des requêtes sur la base de données

Les requêtes sur les collections peuvent dans certain cas être assez long à retourner le résultat.

Pour augmenter la vitesse d'exécution vous pouvez :

- Ajouter des index sur les collections, c'est la méthode la plus efficace.  
Il suffit de rajouter les index en fonction de la requête que l'on souhaite effectuer je vous invite à consulter la documentation de mongolite, la partie sur les index (<https://jeroen.github.io/mongolite/query-data.html#query-syntax>) vous pouvez aussi utiliser la fonction que j'ai implémenté donc notamment `AddIndex(...)` qui permet de rajouter un index sur une colonne d'une collection
- Privilégié des find consécutif plutôt que d'utiliser des agrégations (<https://jeroen.github.io/mongolite/query-data.html#query-syntax>)  
Par exemple, dans le module de génération de requête, pour faire la jointure entre deux collections, je fais deux requêtes, la première me permet de récupérer les clés et la deuxième de récupérer les données correspondantes au clés de la première requête. Le fait d'avoir rajouté des index rend ces deux opérations beaucoup plus rapides que d'utiliser une agrégation.
- Utiliser les fonctions de type **apply**, **lapply** plutôt que les boucle **for** ou **while**.  
Par exemple, lorsque j'ai dû calculer le nombre de ko pour chaque CAGXXXX. Avec une boucle le traitement pouvait durer jusqu'à 3h, alors qu'avec un lapply, j'ai mis 30 min.  
[insérer bout de code et bout du résultat]

## VII. Description des variables principales

Les deux variables les plus importante de l'application sont la variable app et la variable globalinformation.

### 1. App

C'est la variable commune à tous les utilisateurs de l'application, elle est très utile pour l'administration de l'application et facilite ainsi le débogage. Cette variable se trouve dans le fichier global.R. Les données qu'elle contient sont :

- config : contient les données de configuration de l'application . [inserer fichier config.json ]
- usersTracking : c'est un dataframe qui contient les données personnel des utilisateurs connecter [inserer img du dataframe]  
users : c'est une liste dans laquelle chaque élément est un utilisateur connecte (exemple : app\$user[“root@g.c”] ) et ces éléments contiennent les données des modules qui sont unique pour chaque utilisateur (voir plus bas : cette partie est plus détaillé dans globalinformation)  
[inserer img des données contenu dans la liste]

```
> names(app$users$`root@g.c`)
[1] "user"
[2] "typeColumnCollections"
[3] "catalogCollections"
[4] "collections"
[5] "jointuresCollections"
[6] "globalDatabase"
[7] "workDirectory"
[8] "dataFrameLesUtilisateurs"
[9] "usersCollectionsDelete"
[10] "projectsCollections"
[11] "Console"
[12] "moduleCoverageCollections"
[13] "usersCollections"
[14] "modules"
```

- modules : contient le nom des fichiers qui permet la configuration des modules ( les fichiers de définition des interface, de présentation des module dans le menu et les fichier dans lequel les observation et évènement sont défini )
- config\_df : contient la configuration de chaque module, de cette manier on peut ordonné le menu ,décidé quel module afficher en fonction des droit de l'utilisateur qui se connecte ...

Chaque fois qu'un utilisateur se connecte, on rajoute sa variable globalInformation à app\$user[loginutilisateur] (exemple : app\$user['root@g.c'] ou app\$users`root@g.c` et lorsqu'il se déconnecte, on le supprime de cette liste. De cette manière, la longue de cette variable correspond au nombre d'utilisateur en ligne

## 2. globalInformation

Chaque utilisateur connecter à cette variable, sa structure est commune pour tous les utilisateurs, mais son contenu quant à lui est unique. Cette variable se trouve dans le fichier serverVariables.R. Les données que l'on y retrouve sont :

- user : contient les données de l'utilisateur connecter
- typeColumnCollections , catalogCollections, collections, jointuresCollections, usersCollections, usersCollectionsDelete, projectsCollections, moduleCoverageCollections : ce sont les collection fréquemment utiliser par l'application
- globalDatabase : la base de données principal, dans notre cas c'est **iogold**
- workDirectory : le répertoire de l'utilisateur connecté, c'est ce repertoir qu'il faut utiliser pour sauvegarder ou restaurer les données de l'utilisateur
- Console : c'est cette variable qui est utiliser pour écrire dans la fenêtre des logs. C'est dire que chaque fois que l'on souhaite affiche un message dans les log, il suffit de rajouter a cette variable

```
globalInformation$Console <-  
paste0(globalInformation$Console, "Enregistrement Impossible !!, merci de  
modifier le nom de la requête ou le nom du fichier de sauvegarde\n")
```

[ inserer img de la fenetre log]

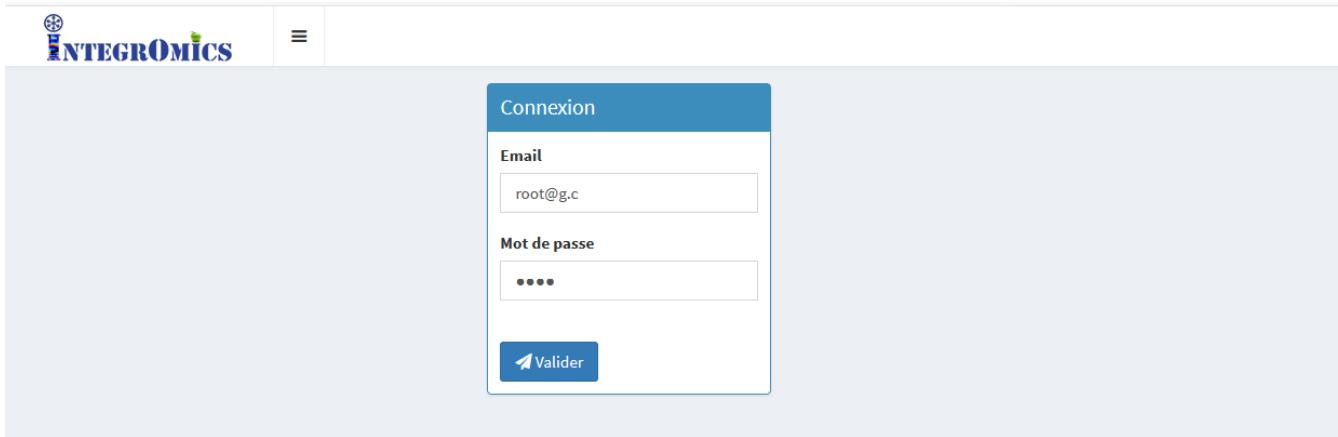
- modules : contient les données utiliser par chaque modules. [inserer bout de code de serverVariable.R

## VIII. Présentation des modules et perspectives d'évolution

### 1. Les principaux modules

#### a. Connexion / login

*Vu de l'interface*



*Description*

Dans ce module se trouve :

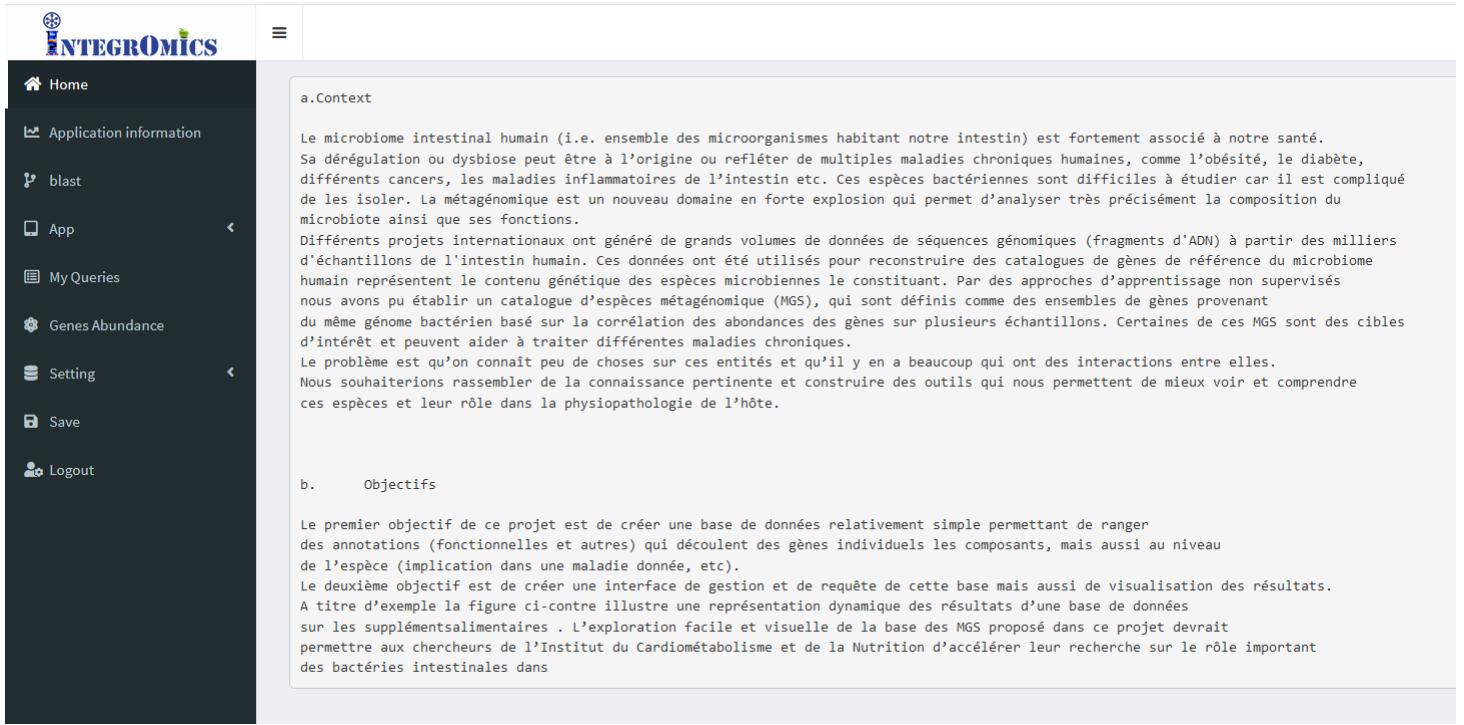
- L'interface de connexion
- Le teste de vérification du mot de passe
- Le choix des modules à afficher en fonction du niveau d'accès des utilisateurs (access\_level)
- La création du menu
- L'ajout du nouvel utilisateur dans la variable app

*Évolution*

- Améliorer l'interface de connexion en affichant des messages explicite en cas de refus d'accès
- Rajouter de nouvelles fonctionnalités
- Modifier le design

#### b. Accueil / Home

*Vu de l'interface*



**a. Context**

Le microbiome intestinal humain (i.e. ensemble des microorganismes habitant notre intestin) est fortement associé à notre santé. Sa dérégulation ou dysbiose peut être à l'origine ou refléter de multiples maladies chroniques humaines, comme l'obésité, le diabète, différents cancers, les maladies inflammatoires de l'intestin etc. Ces espèces bactériennes sont difficiles à étudier car il est compliqué de les isoler. La métagénomique est un nouveau domaine en forte explosion qui permet d'analyser très précisément la composition du microbiote ainsi que ses fonctions.

Différents projets internationaux ont généré de grands volumes de données de séquences génomiques (fragments d'ADN) à partir des milliers d'échantillons de l'intestin humain. Ces données ont été utilisées pour reconstruire des catalogues de gènes de référence du microbiome humain représentant le contenu génétique des espèces microbiennes le constituant. Par des approches d'apprentissage non supervisés nous avons pu établir un catalogue d'espèces métagénomique (MGS), qui sont définis comme des ensembles de gènes provenant du même génome bactérien basé sur la corrélation des abondances des gènes sur plusieurs échantillons. Certaines de ces MGS sont des cibles d'intérêt et peuvent aider à traiter différentes maladies chroniques.

Le problème est qu'on connaît peu de choses sur ces entités et qu'il y en a beaucoup qui ont des interactions entre elles. Nous souhaiterions rassembler de la connaissance pertinente et construire des outils qui nous permettent de mieux voir et comprendre ces espèces et leur rôle dans la physiopathologie de l'hôte.

**b. Objectifs**

Le premier objectif de ce projet est de créer une base de données relativement simple permettant de ranger des annotations (fonctionnelles et autres) qui découlent des gènes individuels les composants, mais aussi au niveau de l'espèce (implication dans une maladie donnée, etc).

Le deuxième objectif est de créer une interface de gestion et de requête de cette base mais aussi de visualisation des résultats. A titre d'exemple la figure ci-contre illustre une représentation dynamique des résultats d'une base de données sur les suppléments alimentaires . L'exploration facile et visuelle de la base des MGS proposé dans ce projet devrait permettre aux chercheurs de l'Institut du Cardiométabolisme et de la Nutrition d'accélérer leur recherche sur le rôle important des bactéries intestinales dans

## Description

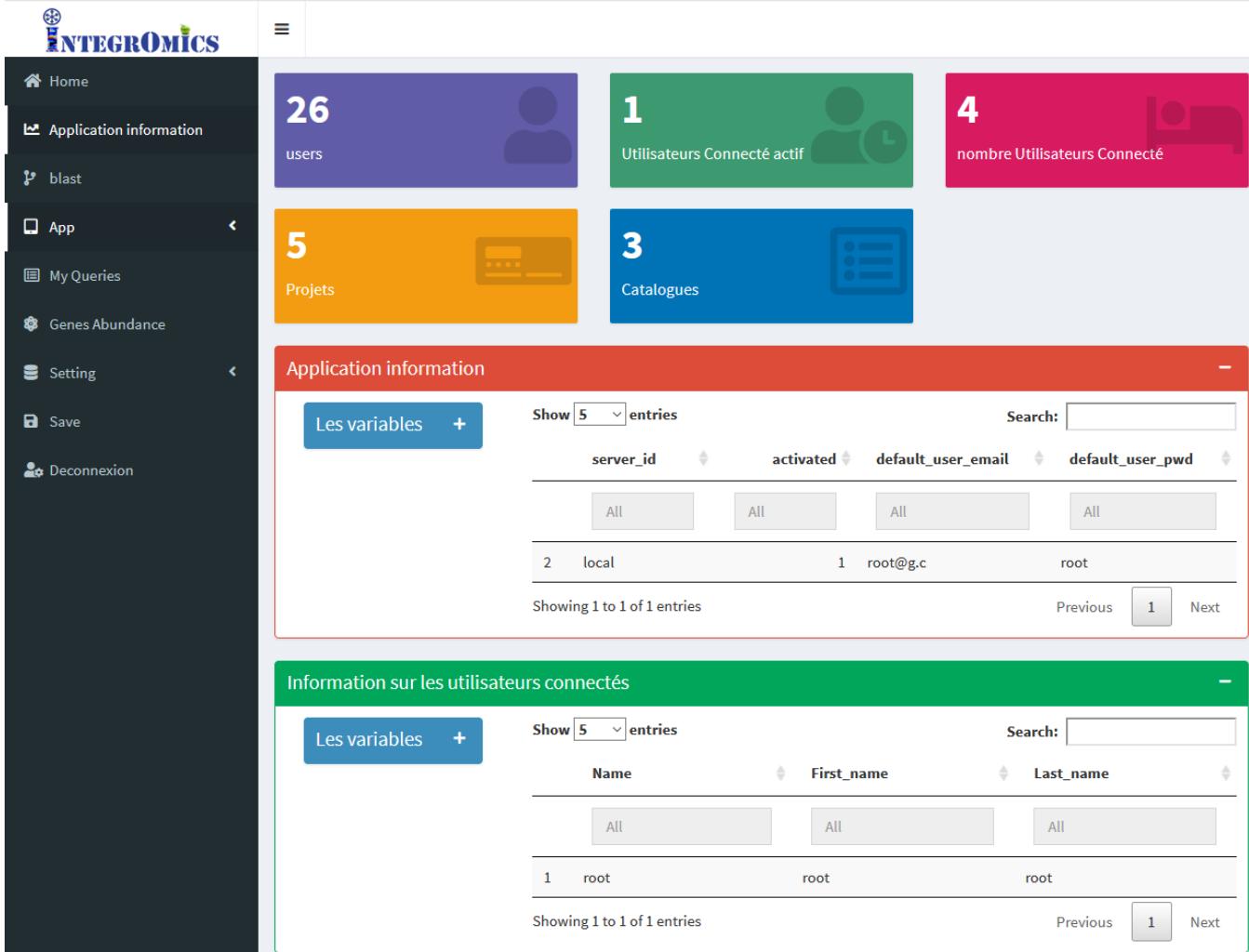
Présente un résumé de ce que fait l'application, à qui elle est destiné et comment l'utiliser

## Évolution

A améliorer en fonction de l'évolution de l'application

### c. Information connexion / connexion information

Vu de l'interface



The screenshot shows the INTEGRONICS application interface with a sidebar and two main content areas.

- Left Sidebar:**
  - Home
  - Application information
  - blast
  - App
  - My Queries
  - Genes Abundance
  - Setting
  - Save
  - Deconnexion
- Top Summary Cards:**
  - 26 users
  - 1 Utilisateurs Connecté actif
  - 4 nombre Utilisateurs Connecté
  - 5 Projets
  - 3 Catalogues
- Application information Module:**

Les variables + Show 5 entries Search: [ ]

server_id	activated	default_user_email	default_user_pwd
All	All	All	All
2 local	1	root@g.c	root

Showing 1 to 1 of 1 entries Previous 1 Next
- Information sur les utilisateurs connectés Module:**

Les variables + Show 5 entries Search: [ ]

Name	First_name	Last_name
All	All	All
1 root	root	root

Showing 1 to 1 of 1 entries Previous 1 Next

#### Description

Dans ce module se trouve quelques informations importantes sur le fonctionnement de l'application

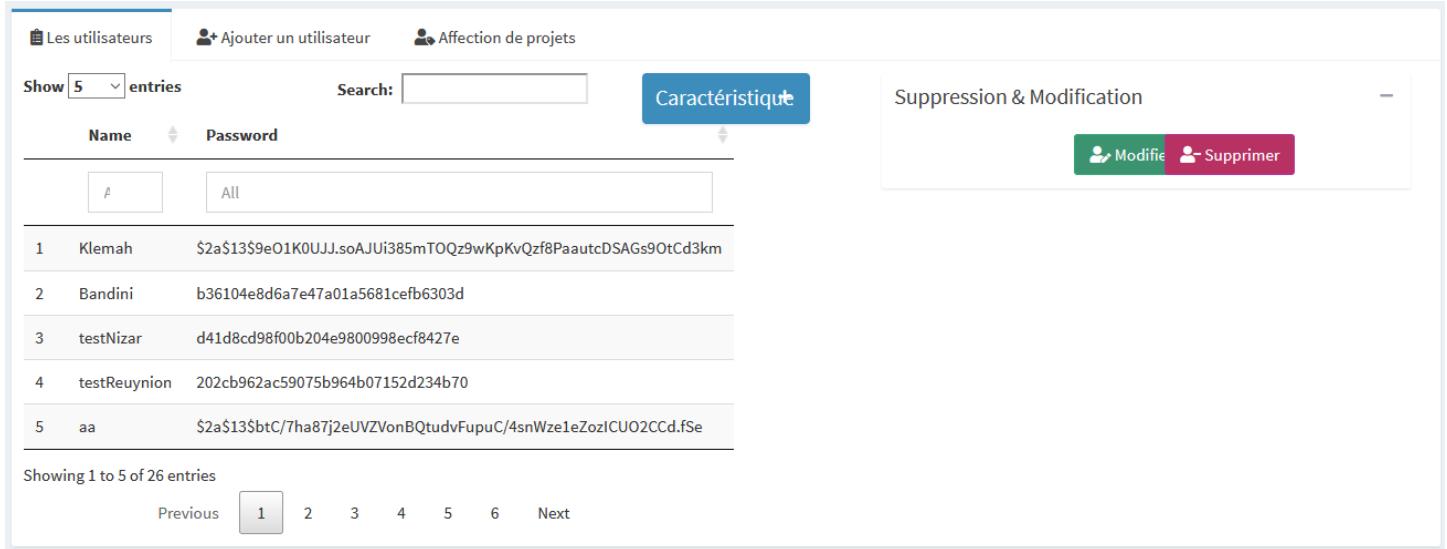
- Le nombre d'utilisateurs connecter
- Le nombre de projets
- Le nombre de catalogue
- Les paramètres de configuration (ceux du fichier config.json )

#### Évolution

- Rajouter des variables réactives ou des boutons afin d'actualiser certaines données
- Modifier le design
- Rajouter de nouvelle statistique
- Vérifier la cohérence de certaines informations

#### d. Gestion des utilisateurs / users

##### Vu de l'interface



Name	Password
1 Klemah	\$2a\$13\$9e01K0UJ.soAJUi385mTOQz9wKpKvQzf8PaauctDSAGs90tCd3km
2 Bandini	b36104e8d6a7e47a01a5681cefb6303d
3 testNizar	d41d8cd98f00b204e9800998ecf8427e
4 testReunion	202cb962ac59075b964b07152d234b70
5 aa	\$2a\$13\$btc/7ha87j2eUVZVonBQtudvFupuC/4snWze1eZozlCUO2CCd.fSe

Showing 1 to 5 of 26 entries

Previous 1 2 3 4 5 6 Next

##### Description

Dans ce module se trouve toute les fonctionnalités qui permet d'administrer les utilisateurs de l'application :

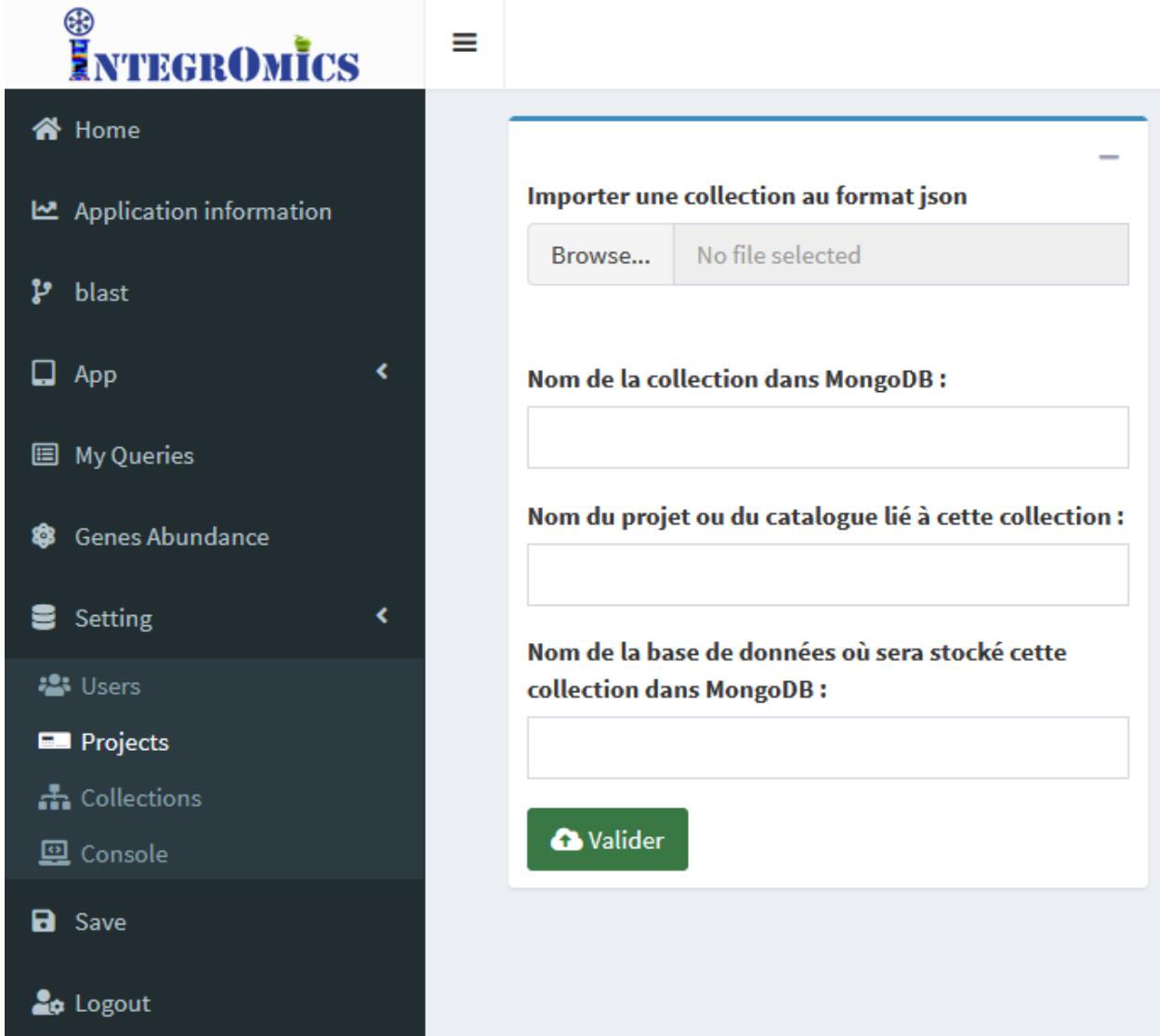
- Consulter la liste des utilisateurs
- Ajouter des utilisateurs
- Supprimer des utilisateurs
- Modifier les données d'un utilisateur
- Affecter des projets à un utilisateur

##### Évolution

- Modifier le design
- Ajouter des boutons ou des variables réactives pour actualiser certaines données

## e. Gestion des projets / projects

*Vu de l'interface*



The screenshot shows the INTEGRONICS application interface. On the left, there is a dark sidebar menu with the following items:

- Home
- Application information
- blast
- App
- My Queries
- Genes Abundance
- Setting
- Users
- Projects
- Collections
- Console
- Save
- Logout

The main content area has a title "Importer une collection au format json". It contains three input fields and a "Valider" button:

- "Browse..." button and "No file selected" label.
- "Nom de la collection dans MongoDB :" input field.
- "Nom du projet ou du catalogue lié à cette collection :" input field.
- "Nom de la base de données où sera stocké cette collection dans MongoDB :" input field.
- A green "Valider" button with a cloud icon.

### Description

Dans ce module, il est possible de rajouter des données, qui ne sont pas très volumineuse, dans la base de données. Ce module est encore à améliorer mais il fonctionne très bien

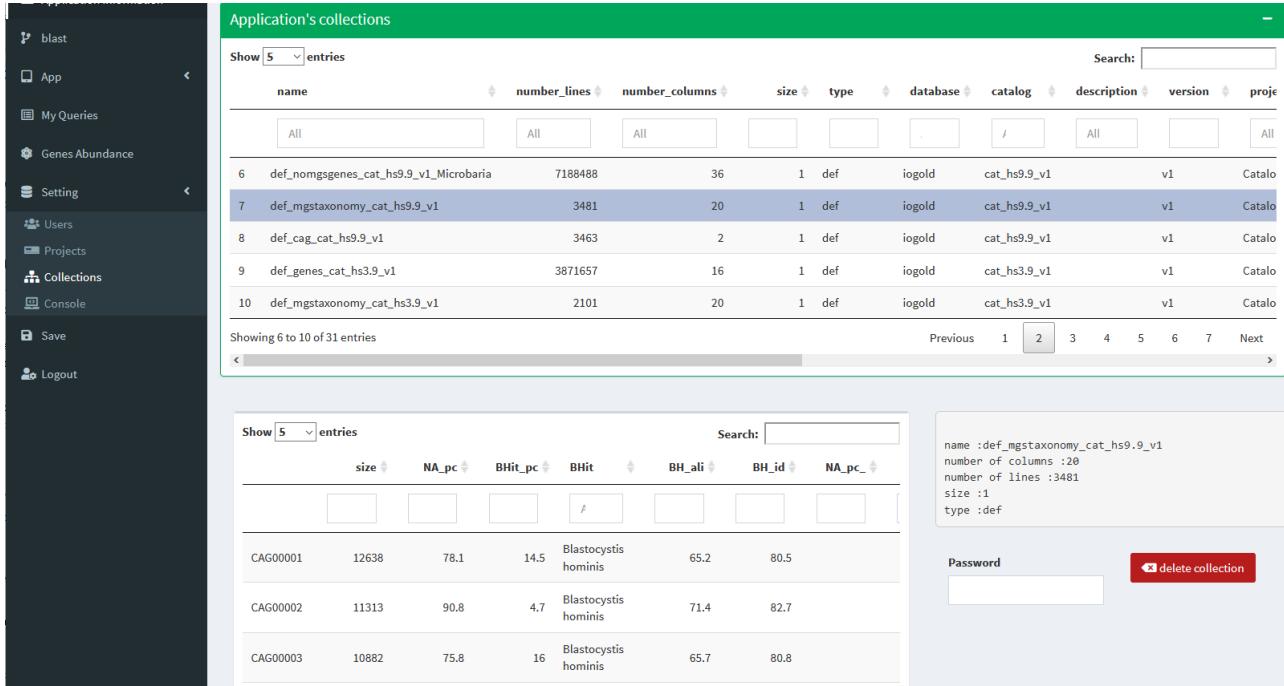
### Évolution

- Définir le type de collection que l'on pourra rajouter via cette interface
- Afficher des informations qui guide l'utilisateur sur le fonctionnement de ce module
- Modifier le design
- Modifier la capacité max des fichiers que l'on peut charger

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

## f. Collections

### Vu de l'interface



	name	number_lines	number_columns	size	type	database	catalog	description	version	proje
6	def_nomsgenes_cat_hs9.9_v1_Microbacteria	7188488	36	1	def	iogold	cat_hs9.9_v1	v1	Catalo	
7	def_mgstanatomy_cat_hs9.9_v1	3481	20	1	def	iogold	cat_hs9.9_v1	v1	Catalo	
8	def_cag_cat_hs9.9_v1	3463	2	1	def	iogold	cat_hs9.9_v1	v1	Catalo	
9	def_genes_cat_hs3.9_v1	3871657	16	1	def	iogold	cat_hs3.9_v1	v1	Catalo	
10	def_mgstanatomy_cat_hs3.9_v1	2101	20	1	def	iogold	cat_hs3.9_v1	v1	Catalo	

Showing 6 to 10 of 31 entries

	size	NA_pc	BHIt_pc	BHIt	BH_ali	BH_id	NA_pc_
CAG00001	12638	78.1	14.5	Blastocystis hominis	65.2	80.5	
CAG00002	11313	90.8	4.7	Blastocystis hominis	71.4	82.7	
CAG00003	10882	75.8	16	Blastocystis hominis	65.7	80.8	

### Description

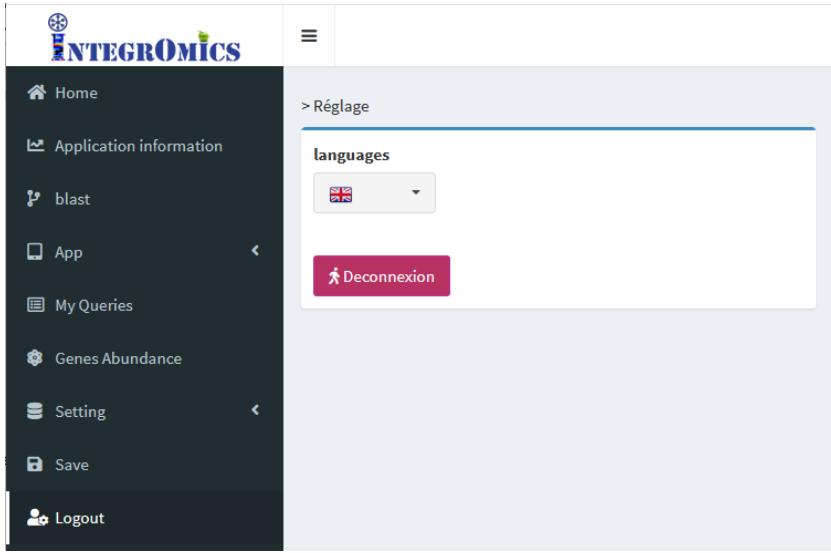
Ce module permet de visualiser toutes les collections utiliser par l'application, afin d'avoir une meilleure visibilité des données que l'on manipule. Il permet également de supprimer des collections

### Évolution

- Modifier le design
- Améliorer les conditions de suppression des collections pour qu'elle soit plus sécurisé
- Dans le cas où la collection appartient à un projet ou un catalogue, il faudra aussi la supprimer de la liste des collections du catalogue ou du projet au quelle elle est liée
- Garder une trace de la personne qui à effectuer la suppression

### g. Déconnexion / Log out

#### Vu de l'interface



The screenshot shows the INTEGRONICS application interface. On the left, there is a dark sidebar menu with the following items:

- Home
- Application information
- blast
- App
- My Queries
- Genes Abundance
- Setting
- Save
- Logout

The main content area has a header "Réglage" and a section titled "languages" with a dropdown menu set to the UK flag. At the bottom of this section is a pink button labeled "Déconnexion".

#### Description

- Permet de se déconnecter de l'application
- Contient l'interface de changement de langue du menu (pour modifier la langue de l'application il faut le faire dans le fichier de configuration de l'application), et de certaine partie (celle défini du coté serveur et qui sont reactive) de certain module

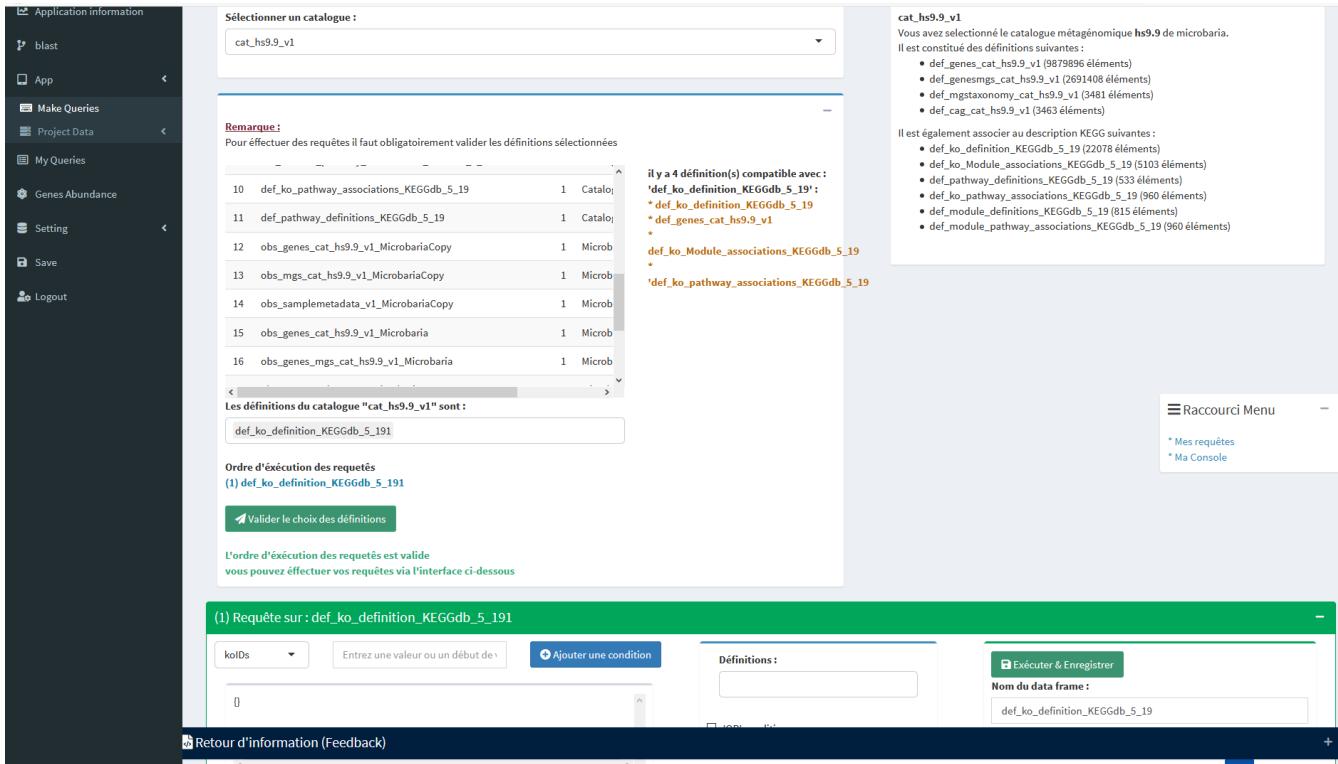
#### Evolution

- Ajouter un bouton de sauvegarde globale avant déconnexion ou implémenter le code de sauvegarde globale à la suite de celui de la déconnexion
- Ajouter une interface qui permettra à l'utilisateur de modifier son mot de passe
- Modifier le design

## 2. Les modules fonctionnels

## a. Générateur de requête / Make Queries

### Vu de l'interface



The screenshot shows the 'Make Queries' section of the ICAN interface. On the left, a sidebar lists options like Application information, blast, App, Make Queries (selected), Project Data, My Queries, Genes Abundance, Setting, Save, and Logout. The main area has a title 'Selectionner un catalogue:' with a dropdown set to 'cat\_hs9.9\_v1'. Below it, a 'Remarque:' section states: 'Pour effectuer des requêtes il faut obligatoirement valider les définitions sélectionnées'. A list of definitions is shown, with one highlighted in orange: 'def\_ko\_definition\_KEGGdb\_5\_191'. To the right, a panel titled 'cat\_hs9.9\_v1' lists 'Il y a 4 définition(s) compatible avec :' followed by four items. At the bottom, a green bar displays '(1) Requête sur : def\_ko\_definition\_KEGGdb\_5\_191' with fields for 'koiDs' and 'Définitions:', and a button 'Exécuter & Enregistrer'.

### Description

Ce module permet de crées des requêtes afin de récupère des données spécifiques.

- Filtrer les données en reliant de nombreuses collections
- Récupère les données des étapes intermédiaires
- Sauvegarder les requêtes générées
- Sauvegarder les données

### Évolution

- Utiliser les données des requêtes pour définir de nouveaux modules destinés à l'analyse de données
- Relire et améliorer le code
- Ajouter une interface pour ajouter des indexées sur les collections

### Guide d'utilisation

Ci-dessous un cas d'utilisation pour vous permettre de prendre en main ce module

#### Étape 1 : sélectionner, ordonner et valider

- Sélectionné le catalogue sur lequel vous souhaitez faire des requêtes

**Sélectionner un catalogue :**

cat\_hs9.9\_v1

▼

**cat\_hs9.9\_v1**

Vous avez sélectionné le catalogue métagénomique **hs9.9** de microbaria.

Il est constitué des définitions suivantes :

- def\_genes\_cat\_hs9.9\_v1 (9879896 éléments)
- def\_genomesgs\_cat\_hs9.9\_v1 (2691408 éléments)
- def\_mgstat taxonomy\_cat\_hs9.9\_v1 (3481 éléments)
- def\_cag\_cat\_hs9.9\_v1 (3463 éléments)

Il est également associé au description KEGG suivantes :

- def\_ko\_definition\_KEGGdb\_5\_19 (22078 éléments)
- def\_ko\_Module\_associations\_KEGGdb\_5\_19 (5103 éléments)
- def\_pathway\_definitions\_KEGGdb\_5\_19 (533 éléments)
- def\_ko\_pathway\_associations\_KEGGdb\_5\_19 (960 éléments)
- def\_module\_definitions\_KEGGdb\_5\_19 (815 éléments)
- def\_module\_pathway\_associations\_KEGGdb\_5\_19 (960 éléments)

- Sélectionnez la ou les collections sur laquelle vous souhaitez faire des requêtes.

**Remarque :**  
Pour effectuer des requêtes il faut obligatoirement valider les définitions sélectionnées

Show	20	entries	
	Search:		
	name	duplicate	project
1	def_genes_cat_hs9.9_v1	1	Catalog
2	def_genomes_cat_hs9.9_v1	1	Catalog
3	def_nomsgenes_cat_hs9.9_v1_Microbaria	1	Catalog
4	def_mgstaxonomy_cat_hs9.9_v1	1	Catalog
5	def_cag_cat_hs9.9_v1	1	Catalog
6	def_module_definitions_KEGGdb_5_19	1	Catalog

Les définitions du catalogue "cat\_hs9.9\_v1" sont :

`def_ko_definition_KEGGdb_5_191 def_genes_cat_hs9.9_v11`

Ordre d'exécution des requêtes

(1) `def_ko_definition_KEGGdb_5_191`  
(2) `def_genes_cat_hs9.9_v11`

 Valider le choix des définitions

- Si vous souhaitez faire une requête qui fait intervenir plusieurs collections, il vous suffit de sélectionner les collections dans l'ordre que vous souhaitez, il sauf juste s'assurer que les collections ont une variable de liaison. Pour cela, vous pouvez utiliser les propositions génère automatiquement.

Il y a 6 définition(s) compatible avec :

'def\_genes\_cat\_hs9.9\_v1':

\* def\_genes\_cat\_hs9.9\_v1

\* def\_cag\_cat\_hs9.9\_v1

\*

\* def ko\_Module\_associations KEGGdb 5.19

3

## def ko\_pathway\_associations KEGGdb 5.1

1

- Une fois votre choix fait, valider le. Une interface vous permettant de définir vos requêtes pour chaque collection sélectionnée s'affiche

(1) Requête sur : def\_ko\_definition\_KEGGdb\_5\_191

(2) Requête sur : def\_genes\_cat\_hs9.9\_v1

Définitions :

Exécuter & Enregistrer

Nom du data frame :

Nombre d'éléments trouvé : -1

Nombre d'éléments trouvé : -1

Raccourci Menu

Mes requêtes  
Ma Console

Exécution Rapide

## Étape 2 : personnaliser sa requête

Exemple de requête sur la collection def\_genes\_cat\_hs9.9\_v1 :

- Requête conjonctive A Et B

GeneLength

5,335 - 8,776

Ajouter une condition

CohortOrigin

usa

Ajouter une condition

Définitions :

GeneLength = [ 5335 - 8776 ]

CohortOrigin ~='usa'

Exécuter & Enregistrer

Nom du data frame :

def\_genes\_cat\_hs9.9\_v1

'OR' condition

- Requête disjonctive : A Ou B

Il suffit simplement de cocher la case 'OR'

### Définitions :

GeneLength = [ 5335 - 8776 ]

CohortOrigin ~= 'usa'

'OR' condition

- Requête avancée

Si vous souhaitez faire plusieurs requêtes sur la même collection, vous pouvez :

- Méthode 1 : Précise le nombre de fois que vous souhaitez faire des requêtes sur cette collection

Pour cela vous devez simplement modifier la colonne duplique du data frame des collections et ensuite vous valider votre choix.

**Pour modifier une colonne, faites une double clique**

name	duplicate	project
1 def_genes_cat_hs9.9_v1	1	CatalogAnnotation_hs
2 def_genomes_cat_hs9.9_v1	1	CatalogAnnotation_hs
3 def_nomsgenes_cat_hs9.9_v1_Microbaria	1	CatalogAnnotation_hs
4 def_mgstaxonomy_cat_hs9.9_v1	1	CatalogAnnotation_hs
5 def_cag_cat_hs9.9_v1	1	CatalogAnnotation_hs
6 def_module_definitions_KEGGdb_5_19	1	CatalogAnnotation_hs

Les définitions du catalogue "cat\_hs9.9\_v1" sont :

def\_genes\_cat\_hs9.9\_v11

	name	duplicate	project
1	def_genes_cat_hs9.9_v1	2	CatalogAnnotation_hs
2	def_genomes_cat_hs9.9_v1	1	CatalogAnnotation_hs
3	def_nomsgenes_cat_hs9.9_v1_Microbaria	1	CatalogAnnotation_hs
4	def_mgstaxonomy_cat_hs9.9_v1	1	CatalogAnnotation_hs
5	def_cag_cat_hs9.9_v1	1	CatalogAnnotation_hs
6	def_module_definitions_KEGGdb_5_19	1	CatalogAnnotation_hs
<			>
Les définitions du catalogue "cat_hs9.9_v1" sont :			
<a href="#">def_genes_cat_hs9.9_v11</a> <a href="#">def_genes_cat_hs9.9_v12</a>			

- Méthode 2 : Cliquer sur le bouton **New** pour générer une nouvelle interface de requête sur la collection

Exécuter & Enregistrer

**Nom du data frame :**

New

Je vous recommande la méthode 1, car elle marche très bien et surtout elle ne réinitialise pas les requête déjà effectuer.

### Étape 3 : visualiser les résultats

```
{ "GenelD":{ "$in": [1824,6989,10908,12225,12989,2310,6771,15252,1458,4016,5647,9770,13984,12529,2034,7433,11514,1354, "$regex": "USA", "$options": "i"}, "CohortOrigin": { "$exists": true }}
```

Nombre d'éléments trouvé : 1056

## Résultat de ma requête

Sélectionner une trame de données enregistrer :

def\_genes\_cat\_hs9.9\_v1

Search:

ID	GeneName	GeneLength	GeneCompletenessStatus	CohortOrigin	TaxID
All	All	All	All	All	All
300	706846339-stool1_revised_scaffold53487_1_gene72330	10971	Complete	USA	unki
316	764143897-stool1_revised_scaffold13752_2_gene76382	10914	Complete	USA	unki
338	159268001-stool2_revised_scaffold22807_1_gene98966	10839	Lack 3-end	USA	unki
341	765640925-stool1_revised_scaffold15488_1_gene87624	10821	Lack 3-end	USA	unki
354	763536994-stool2_revised_scaffold8005_3_gene135320	10794	Complete	USA	unki

Previous 1 2 3 4 5 ... 212 Next

< >

## Étape 4 : sauvegarder ma requête

Seul les requêtes sauvegarder pourront être accessible et manipulable.

Pour consulter la liste des requêtes il faut aller dans l'onglet **My Queries**

### Sauvegarder ma requête

\*Saisir le nom de la requête :

\*Saisir une description de ma :

cette requête nous donne la liste des gènes qui proviennent des USA et qui ont une longueur comprise entre 5563 et 1

\*Saisir le type de la requête :

\*Nom du fichier

 Enregistrer ma requête

Show  entries

Search:

id_collection	collection	requete
All	All	All
1	def_genes_cat_hs9.9_v11	def_genes_cat_hs9.9_v1 { "GeneLength":{ "\$gte" :5563, "\$lte":10975}}
2	def_genes_cat_hs9.9_v12	def_genes_cat_hs9.9_v1 { "GenelD":{ "\$in": [1824,6989,10908,12225,12989,2310,6771,15252,1458,401 "\$regex" : "USA", "\$options" : "i" }, "CohortOrigin":{ "\$exist": true }}

Showing 1 to 2 of 2 entries

Previous  Next

**Astuce :** Double cliquer sur une cellule du data frame pour la modifier.

## b. Mes requêtes / My queries

Vu de l'interface

The screenshot shows the INTEGRONICS software interface. On the left, there is a sidebar with the following navigation options:

- Home
- Application information
- blast
- App
  - Make Queries
  - Project Data
- My Queries
- Genes Abundance
- Setting
- Save
- Logout

The main area contains two tables. The first table, titled "Actualiser", displays a list of entries with columns "user", "type", and "m". The second table displays a list of entries with columns "id\_collection", "collection", and "requete". Both tables include search bars, sorting options, and pagination controls.

Actualiser		
Show 5 entries Search:		
	user	type
1	root@g.c	type1
3	root@g.c	type2
6	root@g.c	typ
7	root@g.c	type1234
8	root@g.c	type1234

Search: Show 5 entries		
id_collection		collection requete
All	All	
2	def_genes_cat_hs9.9_v11	def_genes_cat_hs9.9_v1 {}
3	def_genes_cat_hs9.9_v11	def_genes_cat_hs9.9_v1 {}
4	def_genes_cat_hs9.9_v11	def_genes_cat_hs9.9_v1 {}
5	def_genes_cat_hs9.9_v11	def_genes_cat_hs9.9_v1 {}

### Description

Ce module permet visualiser les requêtes qui ont été sauvegardés sur le disque.

- Consultation des résultats des requêtes sans les ré-exécuter
- Modification et exécution des requêtes existantes

### Évolution

- Implémenter un début de visualisation dans ce module
- Etendre ce module vers de nouveau module d'analyse

### Guide d'utilisation

Ci-dessous un cas d'utilisation pour vous permettre de prendre en main ce module

- Cliquer sur le bouton actualiser

Actualiser
Show **5** entries
Search:

---

user
type
name

---

---

9	root@g.c	type1234	ma_req1234	ce pe
10	root@g.c	type1234	ma_req1234	ce pe
11	root@g.c	type1234	ma_req1234	ce pe
12	root@g.c	type1234	ma_req1234	ce pe
13	root@g.c	type_gene	gene_USA_et_length_bwt_5563_10975	ce no la gé pr de

<
>

- Sélectionner une requête
  - Modifier la si besoin, il vous suffit de faire un double cliquer sur la colonne à modifier
  - Ré-exécute la si besoin à l'aide du bouton Exécute

Actualiser

13	root@g.c	type_gene	gene_USA_et_length_bwt
14	root@g.c	type_gene	gene_USA_et_length_bwt

Showing 11 to 14 of 14 entries

Show **5** entries Search:

id_collection	collection	requete
All	All	All
13 def_genes_cat_hs9.9_v11	def_genes_cat_hs9.9_v1	{"GeneLength": {"\$gte": 5563, "\$lte": 10975}}
14 def_genes_cat_hs9.9_v12	def_genes_cat_hs9.9_v1	{"GeneID": {"\$in": [1824, 6989, 10908, 12225, 12989, 2310, 6771, 1]}, "SRegex": "USA", "Soptions": "i"}, "CohortC"

Showing 1 to 2 of 2 entries Previous 1 Next

Partager Exécuter

Selectionner un data frame :

def\_genes\_cat\_hs9.9\_v1

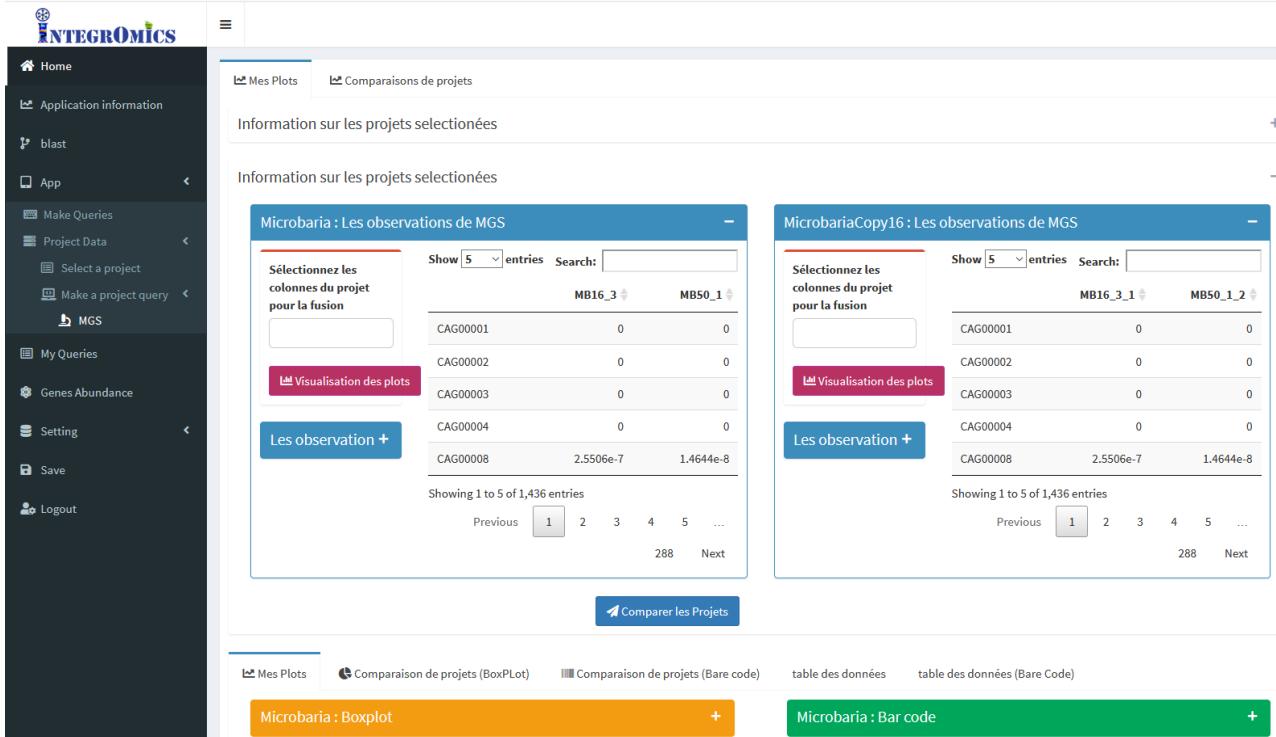
Show **5** entries Search:

GenelD	GeneName	GeneLength	GeneCompletenessStatus	CohortOrigin	TaxonomicPI
	All	All	All	All	All
706846339-stool1_revised_scaffold53487_1_gene72330	1300	706846339-stool1_revised_scaffold53487_1_gene72330	10971	Complete	USA
764143897-stool1_revised_scaffold13752_2_gene76382	1316	764143897-stool1_revised_scaffold13752_2_gene76382	10914	Complete	USA

Page 56 | 92

### c. Données des projets / Project data

#### Vu de l'interface



The screenshot shows the INTEGRONICS web application interface. On the left, a dark sidebar menu includes: Home, Application information, blast, App (with sub-options: Make Queries, Project Data, Select a project, Make a project query, MGS), My Queries, Genes Abundance, Setting, Save, and Logout. The main content area has tabs for Mes Plots and Comparisons de projets. Under Mes Plots, two tables are displayed side-by-side:

- Microbaria : Les observations de MGS**: Shows abundance data for samples MB16\_3 and MB50\_1 across entries CAG00001 to CAG00008. A "Visualisation des plots" button is present.
- MicrobariaCopy16 : Les observations de MGS**: Similar table for samples MB16\_3\_1 and MB50\_1\_2, also showing abundance data for the same entries.

Below these tables is a "Comparer les Projets" button. At the bottom, there are links for Mes Plots, Comparaison de projets (BoxPlot), Comparaison de projets (Bare code), table des données, and table des données (Bare Code). Two buttons at the bottom are highlighted: "Microbaria : Boxplot" (orange) and "Microbaria : Bar code" (green).

#### Description

Ce module est une première approche de l'analyse et de la visualisation des données. Ils traitent uniquement des données de projets, en particulier celle des abondances de MGS. Il permet notamment de :

- Comparer des projets
- Visualiser des plots spécifiques aux analyses médical donc notamment des barres plot, box plot et autre

[Dire ce que fait certains plots]

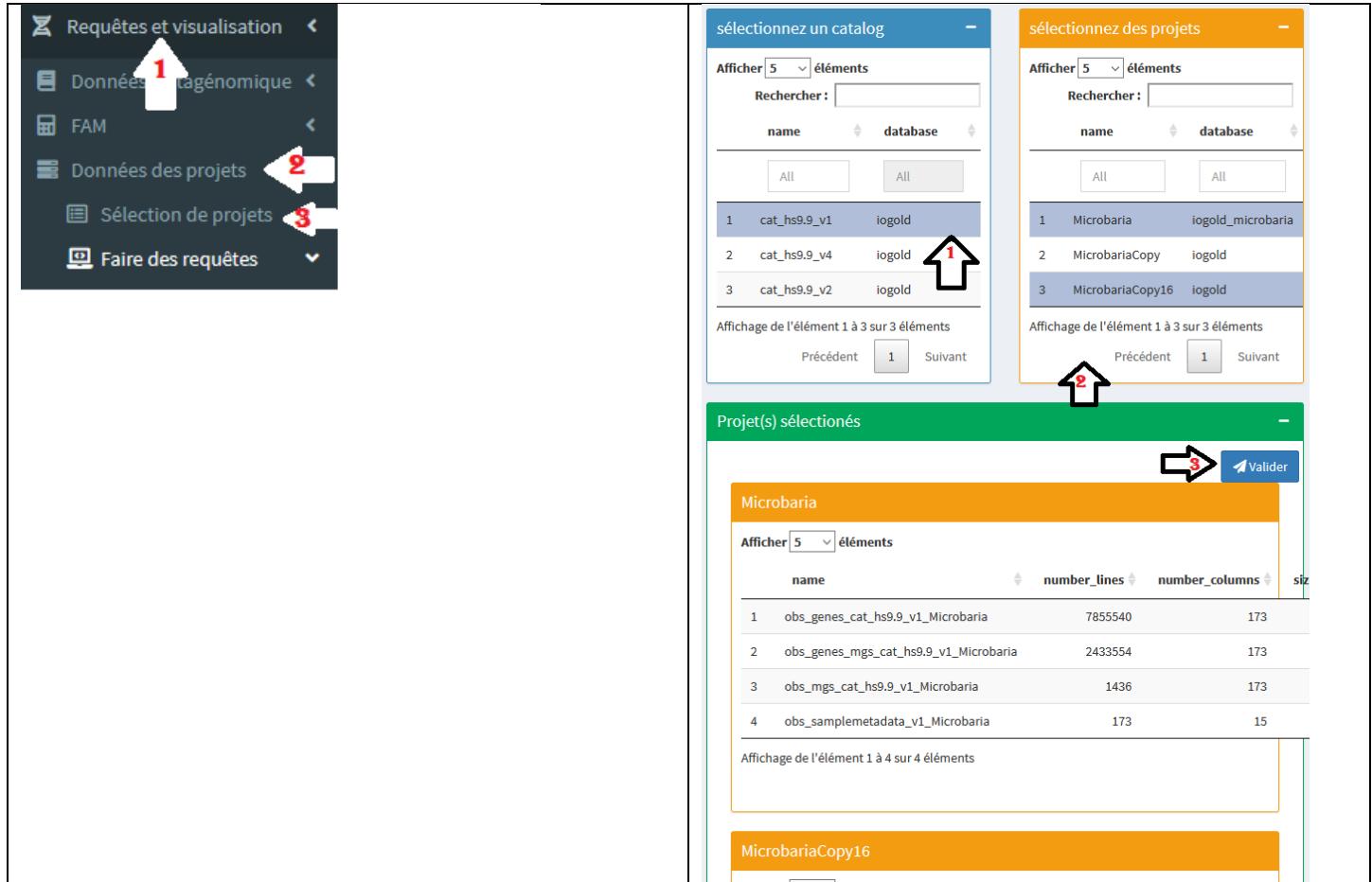
#### Évolution

- Trouver un moyen d'exploiter les données issues du module de génération de requêtes afin d'affiner les analyses
- Modifier le design
- Bien conceptualiser les fonctionnalités de ce module

#### Guide d'utilisation

Ci-dessous un cas d'utilisation pour vous permettre de prendre en main ce module

## Sélectionner des projets



The screenshot shows a user interface for selecting projects across three catalog panels:

- Left Panel:** "Requêtes et visualisation" (1) → "Données d'agénomique" (2) → "Données des projets" (3).
- Top Left Catalog:** "sélectionnez un catalog" → "Afficher 5 éléments".
- Top Right Catalog:** "sélectionnez des projets" → "Afficher 5 éléments".
- Bottom Summary:** "Projet(s) sélectionnés" (3) → "Valider".

Both catalog panels show lists of items with columns for name, database, and other details. The bottom summary panel shows a list of selected projects with a "Microbaria" section.

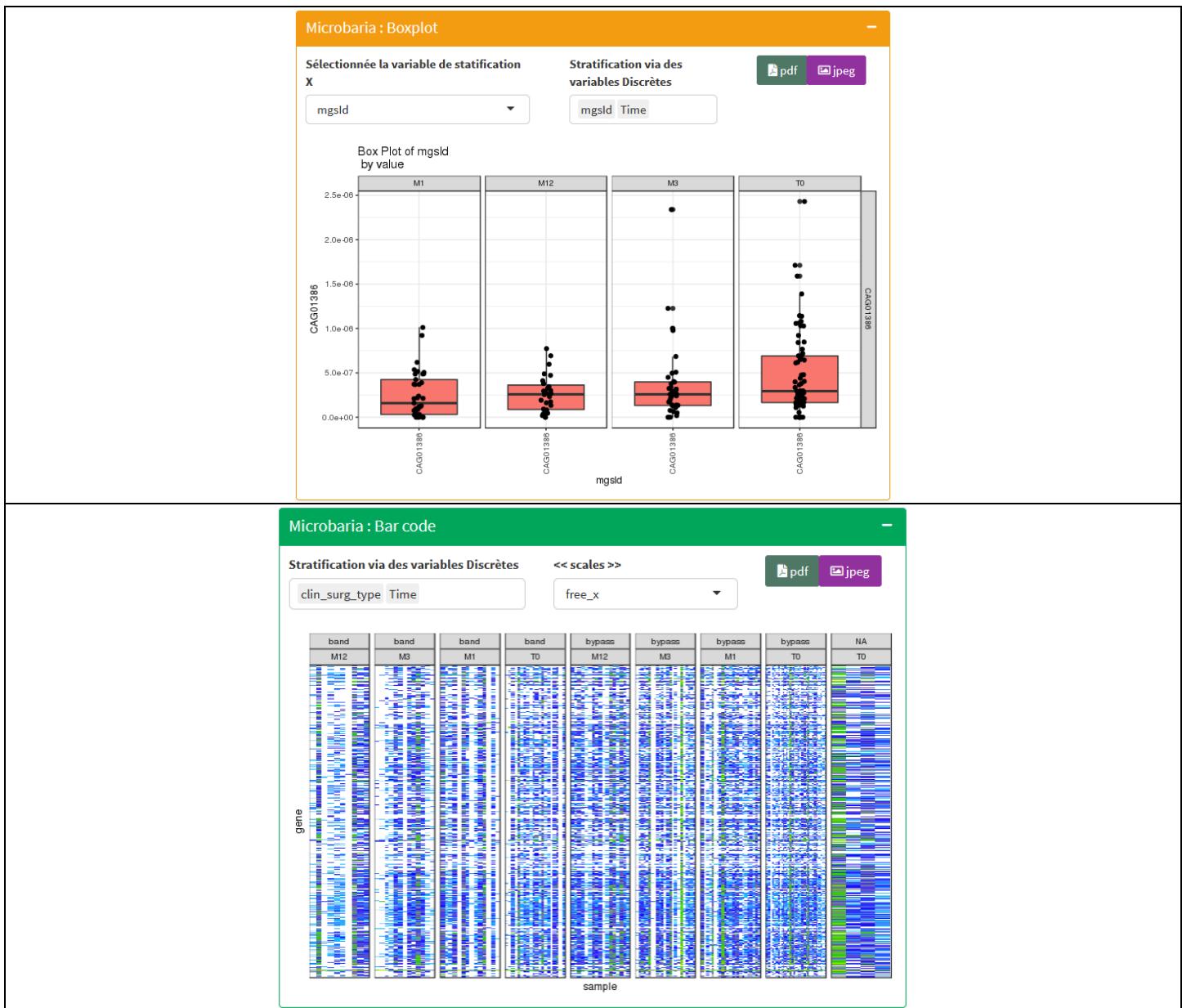
## Analyser les abondances de MGS fusionnés avec les données des projets



The screenshot shows a detailed analysis interface for the "Microbaria" project:

- Left Sidebar:** "Sélectionnez les colonnes du projet pour la fusion" with options: Time, Patient, clin\_surg\_type, down\_3M\_una.
- Right Main Area:** "Show 5 entries" table view with columns MB16\_3, MB50\_1, MB08\_1, MB37\_3.
- Data Table:**

	MB16_3	MB50_1	MB08_1	MB37_3
CAG01383	0	0	0	0
CAG01384	0	0	0	0
CAG01385	0	0	0	0
CAG01386	0	2.7633e-7	7.1826e-7	0
CAG01387	0	0	0	0
- Pagination:** Showing 1,406 to 1,410 of 1,436 entries, with buttons for Previous, Next, and page numbers 1, 281, 282, 283, 288.



## Comparer deux projets

[Mes Plots](#) [Comparaisons de projets](#)

Information sur les projets sélectionnées

Information sur les projets sélectionnées

**Microbaria : Les observations de MGS**

Afficher 5 éléments Rechercher :

	MB16_3	MB50_1
CAG00001	0	0
CAG00002	0	0
CAG00003	0	0
CAG00004	0	0
CAG00008	2.5506e-7	1.4644e-8

Affichage de l'élément 1 à 5 sur 1,436 éléments

Précédent [1](#) [2](#) [3](#) [4](#)

5 ... 288 Suivant

[Visualisation des plots](#) [Les observations](#)

**MicrobariaCopy16 : Les observations de MGS**

Afficher 5 éléments Rechercher :

	MB16_3_1	MB50_1_2
CAG00001	0	0
CAG00002	0	0
CAG00003	0	0
CAG00004	0	0
CAG00008	2.5506e-7	1.4644e-8

Affichage de l'élément 1 à 5 sur 1,436 éléments

Précédent [1](#) [2](#) [3](#) [4](#)

5 ... 288 Suivant

[Visualisation des plots](#) [Les observations](#)

[Comparer les Projets](#)

[Mes Plots](#) [Comparaison de projets \(BoxPlot\)](#) [Comparaison de projets \(Bar code\)](#) [table des données](#) [table des données \(Bar Code\)](#)

**Microbaria : Boxplot** [+](#)

**MicrobariaCopy16 : Boxplot** [+](#)

**Microbaria : Bar code** [+](#)

**MicrobariaCopy16 : Bar code** [+](#)

[Mes Plots](#) [Comparaisons de projets](#)

Statistique [Statistique](#)

Afficher 5 éléments Rechercher :  [Les observations](#)

	MB16_3	MB50_1
CAG01551	0	0

Affichage de l'élément 1 à 1 sur 1 éléments (filtré de 1,436 éléments au total)

Précédent [1](#) Suivant

[Bar Code](#) [box plot](#)

[Mes Plots](#) [Comparaisons de projets](#)

Statistique

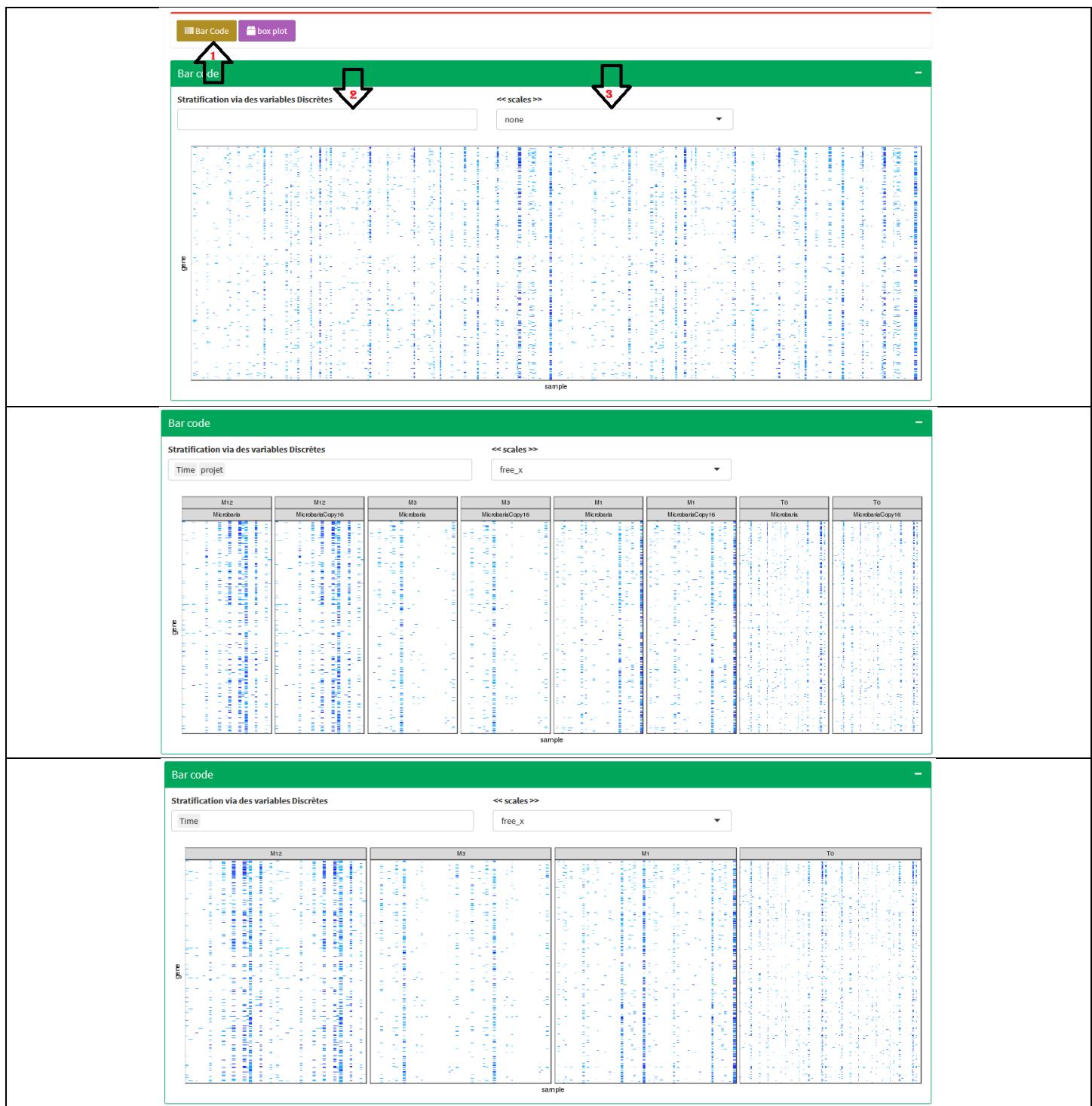
Afficher 5 éléments Rechercher :  [Les variables](#)

	name	number_lines
1	obs_genes_cat_hs9.9_v1_Microbaria	7855540
2	obs_genes_mgs_cat_hs9.9_v1_Microbaria	2433554
3	obs_mgs_cat_hs9.9_v1_Microbaria	1436
4	obs_samplemetadata_v1_Microbaria	173
5	obs_genes_cat_hs9.9_v1_MicrobariaCopy16	7855540

Affichage de l'élément 1 à 5 sur 7 éléments

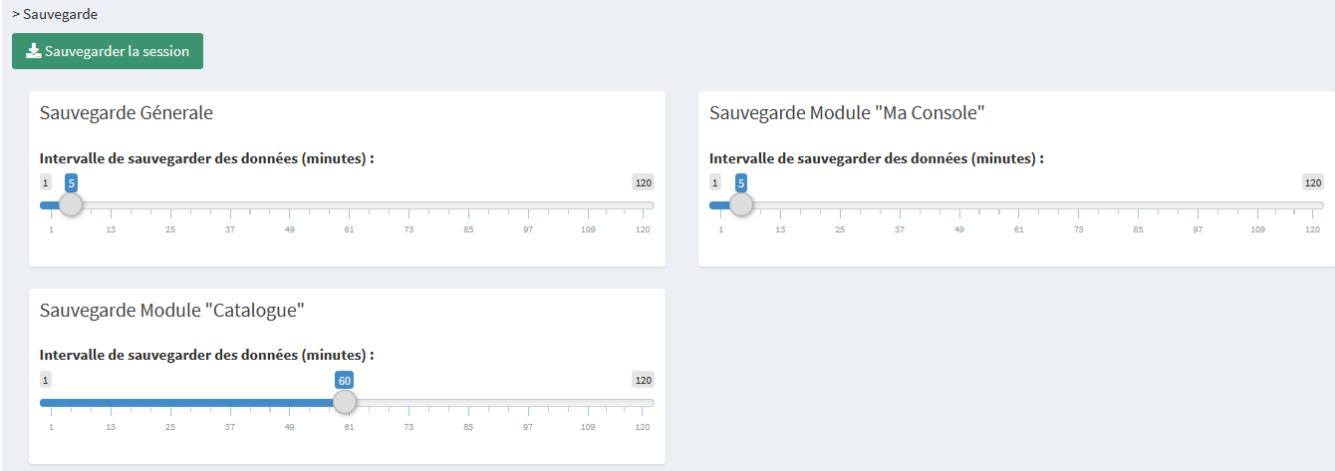
Précédent [1](#) [2](#) Suivant

Les données du nouveau projets :



## d. Sauvegarde /Save

### Vu de l'interface



The screenshot shows the 'Sauvegarde' (Save) section of the ICAN application. It contains three main sections:

- Sauvegarde Générale**: Shows a slider for 'Intervalle de sauvegarder des données (minutes)' (Data save interval (minutes)) ranging from 1 to 120, with a value of 5 selected.
- Sauvegarde Module "Ma Console"**: Shows a slider for 'Intervalle de sauvegarder des données (minutes)' ranging from 1 to 120, with a value of 5 selected.
- Sauvegarde Module "Catalogue"**: Shows a slider for 'Intervalle de sauvegarder des données (minutes)' ranging from 1 to 120, with a value of 60 selected.

### Description

C'est dans ce modules que l'on trouve :

- Les interfaces de sauvegarde périodique des modules
- Le code de récupération des données sauvegarder
- Le code de sauvegarde des données des modules
- Le code de sauvegarde de la variable globalInformation (voir description des variables principal)
- Le code de copie de la variable globalInformation dans la variable app

### Évolution

- Rajouter des boutons de sauvegarde instantané pour chaque module et pour toutes les données de l'application
- Définir des fonctions spécifiques à la restauration complète des données
- Ajouter des checkbox d'activation et de désactivation des sauvegardes
- Modifiées le design
- Ajouter une interface qui permet de restaurer une session sauvegarder

### Guide d'utilisation

Pour sauvegarder votre session utilise le bouton sauvegarder une session

Pour modifier la périodicité de sauvegarde automatique des données, il vous suffit de déplacer le curseur du module concerner

## e. Ma console / Console

*Vu de l'interface*

### Les variable à utiliser :

En fonction du résultat de retour de la console, il faudra sauvegarder le résultat dans la variable qui correspond le mieux.  
`globalInformation$modules$Ma_console$text` : Pour afficher du texte  
`globalInformation$modules$Ma_console$dataFrame` : Pour afficher un data frame  
`globalInformation$modules$Ma_console$plot` : Pour afficher un plot

Ma console : Permet Exécuter du code R

```

1 gi<-list()
2 gi$modules <- list()
3 name_module <- names(globalInformation$modules)
4 for( i in 1:length(name_module))
5 {
6   nmod <- name_module[i]
7   printy(nmod)
8   x<-reactiveValuesToList(globalInformation)
9   gi$modules[[paste0("e",i)]] <- x
10  printy(names()))
11 }
12 printy("fin")
13 printy(length(gi$modules))
14 #printy(names(gi$modules$e3))
15
16
17
18
19

```

Run

Text : print\_text(x)

```

length = tracking
length = root@g.c

length = tracking      length = root@g.c
length = tracking      length = root@g.c
azert
access_level admin catalogSelected connected
dataFrameLesUtilisateurs
dataFrameRequete_Mes_requetes email id

```

DataFrame : print\_dataframe(x)

	MB16_3	MB50_1	MB08_1	MB37_
1000570.HMPREF9966_0917	0	0	* Mes requêtes	* Catalogues
1006551.KOX_14105	0	0		
1006551.KOX_14915	0	0	0	
1006551.KOX_14930	0	0	0	
1028307.EAE_06030	0	0	0	

Showing 1 to 5 of 6,681 entries

Previous 1 2 3 4 5 ... 1337 Next

### Description

Ce module a pour but d'exécuter du code R, Il comprend une console et 4 terminaux de sortie :

- Un terminal pour les plots
- Un terminal pour les data frame
- Un terminal pour le texte
- Et le terminal des logs

### Attention :

Ce module est une faille de sécurité à ne pas négliger. Il faut donc l'utiliser avec précaution.

### Évolution

- Modifier le design afin d'avoir plus d'espace et de lisibilité.

Modifier le paramètre d'auto complétion afin d'avoir plus de choix de fonction et de variables.(voir documentation shinyAce : [https://rdrr.io/cran/dqshiny/man/autocomplete\\_input.html](https://rdrr.io/cran/dqshiny/man/autocomplete_input.html) )

- Rajouter une interface de modification des paramètres d'auto complétion

### Guide d'utilisation

Vous pouvez exécuter du code R, mais pour afficher les résultats vous devez utiliser la fonction

**Printy()**

```
# ** PRINTY_TEXT -----
#' @param obj: a text, dataframe or plot
#' @param stdout: the name of the output to be used (mc: my console module, log:
#console log)
printy_text <- function(obj = "", stdout = "log")
```

### f. Blast

*Vu de l'interface*

The screenshot shows the INTEGRONICS web application interface. On the left is a dark sidebar with navigation links: Home, Application information, blast (selected), App, My Queries, Genes Abundance, Setting, Save, and Logout. The main panel has a header with the INTEGRONICS logo and a 'Load Blast module' button. Below it is a 'Blast search' section with radio buttons for 'blastn' (selected) and 'blastp'. There are input fields for 'evalue' (empty) and 'percent identity' (set to 50). A 'select fasta file' section includes a 'Browse...' button and a message 'No file selected'. A 'fasta selected' section shows an empty input field. Below that is a 'Enter the name of the output file:' input field. Under 'cmd', there is a scrollable text area containing a BLAST command: 'blastn -query /tmp/RtmpSVAvNt /file2dbb27010ba0.fa -db /data/db/biobank /catalogue/hs\_9.9/original\_files/IGC.fa -out'. A dropdown menu for 'outfmt' is set to '0'. At the bottom is a large blue button labeled 'blast!'. The overall layout is clean and modern.

### Description

Dans ce module, je facilite simplement une opération déjà existante, et ceux en définissant une interface plus simple d'utilisation que les commandes linux. En effet, j'utilise les fonctionnalités du module **blast-2.6.0+** afin de :

- Comparer et recherche des séquences de gène similaire à celle passer en entrer au format texte ou via un fichier fasta
- Visualiser les gènes résultants de la requête au format souhaiter
- Jouer avec les options afin d'obtenir des informations personnaliser

### Évolution

- Décrire la commande blast saisie
- Affiche le temps d'exécution
- Modifier le design (label plus explicite, supprimer les bouton inutile)
- Détaille les processus d'ajout
- Ajouter de nouveau option
- Vérifier instantanément le texte passer en entrer
- Définir un fichier de configuration propre à ce module
- Permettre la sélection de plusieurs db
- Permettre l'exécution de plusieurs requêtes pour chaque db sélectionner
- Sauvegarder le data frame du résultat
- Générer des noms de fichier automatique

### Guide d'utilisation

Ci-dessous un cas d'utilisation pour vous permettre de prendre en main ce module

Remarque : ce module est très incomplet, et a été implémenté uniquement pour faire des tests.

C'est pourquoi je vous recommande de remplir tous les champs

- Lors de la première utilisation je vous recommande de recharge le module, pour cela vous pouvez utiliser ce bouton :
 

Load Blast module
- Si vous ne le faites pas vous aurez cette erreur :
 

Error: '/home/ltekam/GIT\_LAB\_test/iogoldg/RShiny/fata\_123' does not exist.
- Compléter tous les champs du formulaire avant de cliquer sur le bouton blast, sinon cela ne fonctionnera pas ou engendrera des erreurs

**Résultat format texte :**

Reference: Zheng Zhang, Scott Schwartz, Lukas Wagner, and Webb Miller (2000), "A greedy algorithm for aligning DNA sequences", J Comput Biol 2000; 7(1-2):203-14.

Database: IGC.fa  
9,879,896 sequences; 7,436,156,055 total letters

Query= eco:b0775 K01012 biotin synthase [EC:2.8.1.6] | (RefSeq) bioB;  
biotin synthase (N)

Length=1041

Score (Bits)	E Value
Sequences producing significant alignments:	
MH0014_GL0111028 [gene] locus=scaffold903_8:91042:92082:-[Complete]	1757 0.0
469595.CSAG_00561 protein_id="ZP_04561231.1" GI="237730750" prod...	1070 0.0
MH0277_GL0004481 [gene] locus=scaffold3692_2:2:1012:-[Lack 3'-end]	1027 0.0
T2D-108A_GL0004668 [gene] locus=scaffold107191_1:27473:28513:+[C...	968 0.0
S07522.KPK_3773 protein_id="Y_P_002239592.1" gene="bioB" GI="2065...	959 0.0
MH0260_GL0024994 [gene] locus=scaffold41307_13:6154:7170:+[Compl...	957 0.0
ED13A_GL0080247 [gene] locus=scaffoldd45954_1:43912:44928:+[Compl...	905 0.0
MH0415_GL0232806 [gene] locus=scaffold99662_2:2:550:-[Lack both ...	798 0.0
V1.CD3-0-PT_GL0031393 [gene] locus=scaffold1092_11:875:1912:-[Co...	486 2e-134

>MH0014\_GL0111028 [gene] locus=scaffold903\_8:91042:92082:-[Complete]  
Length=1041

Score = 1757 bits (951), Expect = 0.0  
Identities = 1011/1041 (97%), Gaps = 0/1041 (0%)  
Strand=Plus/Plus

Query 1 ATGGCTCACCGCCCACGCTGGACATTGTCGCAAGTCACAGAATTATTTGAAAAACCGTTG 60  
 |||||||  
 Sbjct 1 ATGGCTCACCGCCCACGCTGGACATTGTCGCAAGTCACAGAATTATTTGAAAAACCGTTG 60  
 |||||||  
 Query 61 CTGGATCTGCTGTTGAAGGCCAGCAGGTGCATGCCAGATTTGATCCTCGTCAGGTG 120  
 |||||||  
 Sbjct 61 CTGGATCTGCTGTTGAAGGCCAGCAGGTGCATGTCAGCATTTGATCCTCGTCAGGTG 120  
 |||||||  
 Query 121 CAGGTCACTTGCTGCTGATTAAGACGGAGCTTGTCCGGAAAGATTGCAAATACTGC 180  
 |||||||  
 Sbjct 121 CAGGTCACTTGCTGCTGATTAAGACGGAGCTTGTCCGGAAAGATTGCAAATATTGC 180  
 |||||||  
 Query 181 CCGCAAAGCTCGGCTACAAAACCGGGCTGGAAGCCGAGCGTTGATGGAAGTTGAACAG 240  
 |||||||  
 Sbjct 181 CCGCAAAGCTCGGCTACAAAACCGGGCTGGAAGCCGAGCGTTGATGGAAGTTGAACAG 240

## Résultat Format data frame :

outfmt

6

blast!

eco:b0775	MH0014_GL0111028	97.118	1041	30	0	1	1041	1	1041	0.0	1757
eco:b0775	469595.CSAG_00561	85.672	1019	142	4	16	1032	16	1032	0.0	1070
eco:b0775	MH0277_GL0004481	85.119	1008	146	4	26	1031	2	1007	0.0	1027
eco:b0775	T2D-108A_GL0004668	84.545	977	151	0	1	977	1	977	0.0	968
eco:b0775	S07522.KPK_3773 83.317	1043	170	4	1	1041	1	1041	0.0	959	
eco:b0775	MH0260_GL0024994	83.710	1019	160	6	26	1041	2	1017	0.0	957
eco:b0775	ED13A_GL0080247 82.953	1009	166	6	26	1031	2	1007	0.0	905	
eco:b0775	MH0415_GL0232806	92.896	549	39	0	271	819	1	549	0.0	798
eco:b0775	V1.CD3-0-PT_GL0031393	76.923	858	196	2	42	898	42	898	2.09e-134	486

## IX. Ajout de nouveaux modules

### Étape 1

Compiler et Exécuter la fonction newModule(moduleName), Qui se trouve dans le fichier ...../RShiny/R/PackageForAdmin /MesFonctionAdmin.R ( voir arborescence des fichiers de l'application)

Une fois exécuter, Le module est créé et rajouter à l'application. [inserer img menu]

### Étape 2

Il ne vous reste qu'à développer les fonctionnalités du module créé, le code permettant de designer l'interface doit être place dans uiModuleName.R, et celui permettant de définir les évènements dans serverModuleName.R

### Étape supplémentaire

Si le module créer utilisera de nouvelles librairies autre que celle spécifier dans global information, il faudra les rajoutées.

Si l'on souhaite sauvegarder certaines données du module, il faudra définir une nouvelle structure propre à ce module (voir fichier ..../RShiny/server/serverVariables.R) de cette manière, la nouvelle structure sera rajouté à la liste des variables de l'application accessible via **globalInformation** et via la variable globale **app** après sauvegarde automatique ou manuel. [inserer bout de code de serverVariable.R]

Afin de respecter la structuration de l'application. Le code qui permettra de sauvegarder vos données doit être écrit dans le module de sauvegarde [inserer img arborescence fichier et bout de code du module sauvegarde]. Toutefois, ce code peut être mi n'importe où dans n'importe quel fichier mais il est préférable qui soit dans le même fichier que la gestion de sauvegarde des autres modules.

On pourrait également rajouter une fenêtre log comme dans certain module (voir module console ou Requete\_et\_visualisation/Les\_Genes) [inserer img log et bout de code]

## X. Astuce et conseil

### 1. Astuce

Les noms de variables :

Afin d'éviter d'avoir des variables avec le même nom (ce qui bloquera l'exécution de l'application) [inserer img de l'appli bloqué]

Je vous recommande de rajouter un suffixe au nom de vos variables. Par exemple pour la variable nombre du module abc vous pouvez plutôt écrire nombre\_abc. Pour ma part j'ai utilisé une syntaxe de déclaration assez longue mais explicite [inserer des bout de déclaration de variables]

Script volumineux :

Il y a plusieurs manières de procéder si votre script devient assez long :

- Déplacer une partie du code dans un autre fichier (voici le lien qui explique comment faire)
- Découper son code dans deux fichiers
- Ajouter des commentaires sous forme de menu

Pour avoir d'autres astuces vous pouvez consulter ce blogue qui est assez intéressant.

<https://deanattali.com/blog/advanced-shiny-tips/>

Débogage du code :

### 2. Conseil

Les problèmes de développement :

Je pense avoir rencontré et résolu de nombreux problèmes lors du développement. C'est pourquoi je vous invite, si jamais vous rencontrer un problème, à parcourir l'application pour voir si je n'ai pas déjà développé ce que vous recherchez. Vous pouvez également consulter les liens que j'ai mis à votre disposition en annexe.

Librairie existante :

Avant de commencer à développer des fonctionnalités, rechercher si elle n'existe pas déjà dans une autre librairie. Par exemple il y a des librairies que j'ai rajouté juste pour utiliser une ou deux fonctions qui m'auraient pris du temps à développer et n'aurais pas forcément été plus performante

## XI. Perspective d'évolution

### 1. Niveau applicatif

Ils pourront par exemple :

- Implémenter de nouveaux modules de visualisation de données en utilisant les résultats des requêtes déjà effectuées
- Ajouter des modules complètement indépendants du reste du projet afin de centraliser les opérations des chercheurs

Il y a plein de possibilités cela dépendra des besoins des chercheurs.

### 2. Niveau structuration du code

A mon avis il faudra probablement revoir la structuration des données que j'ai utilisé pour effectuer certains traitements, en particulier celle du module de création de requêtes, qui malgré le fait qu'elle tourne comme il faut pourrais être moins complexe.

### 3. Niveau interface

Au niveau de l'interface, elle pourrait être modifiée en utilisant des librairies plus pointues, pour ma part j'ai commencé par utilisé les interfaces par défauts de **shiny**, puis j'ai rajouté les fonctionnalités de **shinydashboard** et ensuite j'ai découvert une nouvelle librairie que je n'ai malheureusement pas eu le temps d'exploiter, il s'agit de :

- shinydashboardplus
- semantic.dashboard

Cette partie est mieux explique dans le rapport technique.

### 4. Niveau base de données

Les possibilité d'optimisation du requêtage de la base de données sont assez nombreuses, j'ai optimalisé les requêtés de base (celle où l'on effectuer la recherche avec une condition exemple, la liste des gènes qui ont pour id inclus dans un certain intervalle), mais si l'on connaît sur quelle(s) attribut(s) les requêtes sont généralement lancer, il serait possible améliore la vitesse de retour des données résultantes.

### 5. Niveau sauvegarde des données

Il faudrait trouver un répertoire sécurisé pour la sauvegarde des données et automatiser la sauvegarde de ce répertoire dans un autre afin de se protéger d'une éventuelle panne ou destruction de données.

## 6. Niveau Conception

Avant de continuer ce projet, ou au moins avant d'ajouter un nouveau module, il faudrait bien conceptualiser la ou les tâches à effectuer, concevoir les interfaces, étudier les cas les plus basiques et bien déterminer le point d'arrivée que ce soit au niveau développement ou au niveau du résultat à obtenir.

## 7. Niveau fonctionnalité

On pourrait implémenter de nouvelles fonctionnalités qui ne sont pas forcément liées au besoin des chercheurs mais qui permettrait tout de même d'avoir une meilleure vision sur l'évolution de l'application, ou qui permettraient au administrateur de mieux gérer les utilisateurs. On pourrait par exemple ajouter une interface de visualisation de statistique d'utilisation de l'application par les utilisateurs (connexion, temps de travail, nouveau arrivant ...) j'ai commencé à développer une toute petite partie de ce type de fonctionnalité, qui est disponible sur l'application mais reste à améliorer.

Il y a de nombreuses autres fonctionnalités que l'on pourrait implémenter :

- Ajout des logs pour chaque module
- Spécifications des variables à sauvegarder (sous forme de checkbox par exemple) + ajout d'un bouton de sauvegarde direct

## XII. Mise en production de l'application

Afin de publier l'application ou de la mettre à jour après développement d'une nouvelle fonctionnalité il faut suivre les étapes suivantes :

1. go to [icr2.intgromics.fr](http://icr2.intgromics.fr) in ssh
2. cd /home/Itekam/GIT\_LAB\_test/iogoldg
3. rsync --append --progress --partial -azvv RShiny/\* [Itekam@portal.integromics.fr](mailto:Itekam@portal.integromics.fr):/share/apps/iogold/iogoldapp/
4. connect ssh to the portal ssh [Itekam@portal.integromics.fr](mailto:Itekam@portal.integromics.fr)
5. cd /share/apps/iogold/iogoldapp/
6. chmod -R 770 \*
7. chgrp -R Itekam:meteor \*
8. go to page <http://iogold.intgromics.fr/>

Remarque :

- Seul un administrateur peut mettre l'application en ligne
- Dans les étapes ci-dessus il faudra changer mes identifiants par celle du nouveau développeur

## XIII. Liens utile

### 1. Documentations et astuce

Documentation mongolite

<https://github.com/jeroen/mongolite>

<https://blog.exploratory.io/an-introduction-to-mongodb-query-for-beginners-bd463319aa4c>

Débuter avec shiny

<https://shiny.rstudio.com/tutorial/>

Débuter avec shinydashboard

<https://rstudio.github.io/shinydashboard/structure.html#boxes>

<https://www.r-pkg.org/pkg/shinyjs>

ShinyAce

<https://github.com/trestletech/shinyAce>

<https://cran.r-project.org/web/packages/shinyAce/readme/README.html>

[https://rdrr.io/cran/dqshiny/man/autocomplete\\_input.html](https://rdrr.io/cran/dqshiny/man/autocomplete_input.html)

Documentation datatable

<https://github.com/rstudio/DT/pull/480>

<https://github.com/rstudio/DT/releases>

Démo blast

<https://2-bitbio.com/2017/06/running-blast-in-shiny-web-application.html>

<https://github.com/larsgr/RLinuxModules>

Icônes et couleur

<https://fontawesome.com/icons?d=gallery&q=data>

[https://www.google.com/search?source=hp&ei=NI4HXdW5NMKLIwTOgIX4Ag&q=color+%23337ab7&oq=colo&gs\\_lpsy-ab.1.0.35i39l2j0j0i131j0l6.1060.1425..2561...0.0..0.56.217.5.....0....1..gws-wiz.....0.AUD1Qr6GkSA](https://www.google.com/search?source=hp&ei=NI4HXdW5NMKLIwTOgIX4Ag&q=color+%23337ab7&oq=colo&gs_lpsy-ab.1.0.35i39l2j0j0i131j0l6.1060.1425..2561...0.0..0.56.217.5.....0....1..gws-wiz.....0.AUD1Qr6GkSA)

Astuce pour développer avec Shiny

<https://deanattali.com/blog/advanced-shiny-tips/>

<https://deanattali.com/blog/advanced-shiny-tips/#shinyjs>

## 2. Aide à la Résolution de problème

<https://www.showmeshiny.com/visualizing-ggplot2/>

[https://github.com/emitanaka/shinycustomloader?fbclid=IwAR3Wkj6zs-cfAv\\_XJw\\_lZyDmpYRaveaY0a3a10sNCgY8sHPt8oH-C49e5Mc](https://github.com/emitanaka/shinycustomloader?fbclid=IwAR3Wkj6zs-cfAv_XJw_lZyDmpYRaveaY0a3a10sNCgY8sHPt8oH-C49e5Mc)

<https://divadnojnarg.github.io/blog/awesomedashboards/>

<https://appslion.com/create-outstanding-dashboards-with-the-new-semantic-dashboard-package/>

<https://www.datanovia.com/en/lessons/subset-data-frame-rows-in-r/>

<https://www.r-bloggers.com/5-ways-to-subset-a-data-frame-in-r/>

<https://openclassrooms.com/fr/courses/4462426-maitrisez-les-bases-de-donnees-nosql/4474606-interrogez-vos-donnees-avec-mongodb>

<https://stackoverflow.com/questions/44780837/r-shiny-bookmarking-dynamically-created-ui>

[https://developers.google.com/youtube/iframe\\_api\\_reference?fbclid=IwAR3f7MEO20HxxJz1CGDHwABiVSScbyFNs0Zxc\\_ghIOKBP8EbDHs2Jzvmk6Zs](https://developers.google.com/youtube/iframe_api_reference?fbclid=IwAR3f7MEO20HxxJz1CGDHwABiVSScbyFNs0Zxc_ghIOKBP8EbDHs2Jzvmk6Zs)

<https://rstudio.github.io/shinydashboard/index.html>

<https://gist.github.com/wch/5436415/>

<https://community.rstudio.com/t/how-to-position-multiple-dynamically-added-ui-elements-in-shiny-r/4036>

[https://rstudio-pubs-static.s3.amazonaws.com/440403\\_2fe4b00a09dd4b268efd6efb353ccad7.html#r%C3%A9cup%C3%A9ration\\_du\\_clic](https://rstudio-pubs-static.s3.amazonaws.com/440403_2fe4b00a09dd4b268efd6efb353ccad7.html#r%C3%A9cup%C3%A9ration_du_clic)

<https://cran.r-project.org/web/packages/mongolite/mongolite.pdf>

<https://reprex.tidyverse.org/>

<https://community.rstudio.com/t/shiny-app-with-dynamic-number-of-datatables/2405>

- <https://stackoverflow.com/questions/8510870/boxplot-from-row-values-in-a-dataframe>
- <https://discuss.analyticsvidhya.com/t/how-to-add-a-column-to-a-data-frame-in-r/3278/2>
- <https://blog.bioturing.com/2018/05/22/how-to-compare-box-plots/>
- <https://github.com/rstudio/shinydashboard/issues/295>
- <https://stackoverflow.com/questions/43404058/starting-shiny-app-after-password-input-with-shinydashboard>
- <http://shiny.rstudio.com/articles/scoping.html>
- [http://r-forge.r-project.org/scm/viewvc.php/\\*checkout\\*/www/datatables-faq.pdf?revision=816&root=datatable&pathrev=830](http://r-forge.r-project.org/scm/viewvc.php/*checkout*/www/datatables-faq.pdf?revision=816&root=datatable&pathrev=830)
- <https://blog.exploratory.io/scrape-data-from-web-pages-50e45b2b150a>
- <https://support.rstudio.com/hc/en-us/articles/207126217-Using-Source-Windows>
- <https://shiny.rstudio.com/articles/running.html>
- <https://stackoverflow.com/questions/37795760/r-shiny-add-weblink-to-actionbutton>
- <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3531135/>
- <https://www.ncbi.nlm.nih.gov/pubmed/?term=belda+and+vallenet>
- <https://www.ncbi.nlm.nih.gov/pubmed/23193269>
- <https://gist.github.com/alain-andre/8150256>
- <https://thinkr.fr/pdf/shiny-french-cheatsheet.pdf>
- <https://stackoverflow.com/questions/32104918/how-to-manually-collapse-a-box-in-shiny-dashboard>
- <https://stackoverflow.com/questions/34832171/using-a-loop-to-create-multiple-data-frames-in-r>
- <https://github.com/rstudio/shiny/issues/532>
- <http://portal.integromics.fr:3838/ltekam/>
- <https://git-scm.com/book/fr/v2/Les-branches-avec-Git-Les-branches-en-bref>
- <https://community.rstudio.com/t/r-shiny-printing-the-console-output-produced-by-an-r-package-to-ui/4926/2>
- <https://deanattali.com/blog/shinyjs-v08/>
- <https://shiny.rstudio.com/reference/shiny/latest/absolutePanel.html>
- <https://github.com/rstudio/shinydashboard/issues/40>
- <https://stackoverflow.com/questions/26368192/how-to-insert-new-line-in-r-shiny-string>

- <https://stackoverflow.com/questions/47478123/how-to-change-background-color-for-textareainput-in-r-shiny>
- <https://stackoverflow.com/questions/36080529/r-shinydashboard-customize-box-status-color>
- <https://cran.r-project.org/web/packages/shinydashboardPlus/vignettes/improved-boxes.html>
- <https://github.com/rstudio/shinydashboard/issues/287>
- <https://stackoverflow.com/questions/32076382/mongodb-how-to-get-max-value-from-collections>
- [https://www.w3schools.com/css/css\\_font.asp](https://www.w3schools.com/css/css_font.asp)
- <https://stackoverflow.com/questions/41563984/set-title-header-in-shiny-dashboard>
- <https://stackoverflow.com/questions/5620885/how-does-one-reorder-columns-in-a-data-frame>
- <https://dplyr.tidyverse.org/reference/distinct.html>
- <http://r.789695.n4.nabble.com/Convert-string-to-list-td830791.html>
- [https://www.datacamp.com/community/tutorials/r-tutorial-apply-family?utm\\_source=adwords\\_ppc&utm\\_campaignid=898687156&utm\\_adgroupid=48947256715&utm\\_device=c&utm\\_keyword=&utm\\_matchtype=b&utm\\_network=g&utm\\_adpostion=1t1&utm\\_creative=229765585183&utm\\_targetid=aud-299261629574:dsa-473406586995&utm\\_loc\\_interest\\_ms=&utm\\_loc\\_physical\\_ms=9056137&gclid=Cj0KCQjw3uboBRDCARIsAO2XcYBQDyidX7JuKcqO38qE-6UkdZAFJLtc8xa67KIKvQpdk-wiOvS6YClAhWdEALw\\_wcB](https://www.datacamp.com/community/tutorials/r-tutorial-apply-family?utm_source=adwords_ppc&utm_campaignid=898687156&utm_adgroupid=48947256715&utm_device=c&utm_keyword=&utm_matchtype=b&utm_network=g&utm_adpostion=1t1&utm_creative=229765585183&utm_targetid=aud-299261629574:dsa-473406586995&utm_loc_interest_ms=&utm_loc_physical_ms=9056137&gclid=Cj0KCQjw3uboBRDCARIsAO2XcYBQDyidX7JuKcqO38qE-6UkdZAFJLtc8xa67KIKvQpdk-wiOvS6YClAhWdEALw_wcB)
- <http://langtest.jp/shiny/cor/>
- <http://theautomatic.net/2018/07/11/manipulate-files-r/>
- <https://stackoverflow.com/questions/8883554/converting-time-format-to-numeric-with-r>
- <https://stackoverflow.com/questions/55682531/r-shinydashboard-mix-of-dynamic-and-static-tabitems-for-various-menuitems>
- <https://stackoverflow.com/questions/2862590/how-to-replace-master-branch-in-git-entirely-from-another-branch>
- <https://www.ardata.fr/post/2019/04/26/share-reactive-among-shiny-modules/>
- <https://stackoverflow.com/questions/47505893/adding-a-vertical-and-horizontal-scroll-bar-to-the-dt-table-in-r-shiny>
- <https://yihui.shinyapps.io/DT-edit/>
- <https://github.com/rstudio/DT/issues/29>
- <https://stackoverflow.com/questions/2900510/r-equivalent-of-select-distinct-on-two-or-more-fields-variables>
- <https://www.youtube.com/watch?v=NmTz-D3cbhU>
- <https://rdrr.io/cran/DT/man/proxy.html>
- <https://github.com/seqan/lambda/wiki/BLAST-Output-Formats>

## Table des matières

Introduction .....	5
I. Présentation du projet .....	6
1. Objectifs du projet .....	6
a. Contexte .....	6
b. Objectifs .....	6
c. Existant .....	7
2. Informations .....	7
3. Connaissances requises .....	7
II. Structuration et description de l'arborescence de l'application .....	8
1. Structure global .....	8
2. Description du contenu des fichiers et dossiers .....	8
a. Config.json .....	8
b. Global.R, ui.R et server.R .....	8
i. Global.R .....	8
ii. Ui.R .....	9
iii. server.R et Server .....	10
❖ Le fichier serveurLesVariables.R : .....	10
❖ Le fichier serveurLesFonctions.R : .....	11
❖ Le fichier serverUpdate_feedback.R : .....	12
c. Modules .....	14
i. Config.json .....	14
ii. uiModules_X.R .....	15
iii. serverModule_X.R .....	15
d. www .....	16
e. R .....	16
i. Arborescence .....	16
ii. Description .....	16
❖ packageForMongoDB : .....	16
❖ PackageForGgplot2 : .....	17
❖ PackageForAdmin : .....	18

❖ ManagingDataBase .....	18
f. user_data .....	19
i. Arborescence .....	19
ii. Description .....	19
III. Configuration de l'application.....	19
1. Exemple de configuration .....	20
2. Description <i>des variables</i> .....	20
IV. Présentation de l'architecture de la base de données et description des différentes collections.....	22
1. Architecture de la base de données .....	22
2. Description des collections .....	23
a. Les collections existantes.....	23
i. Description des données.....	23
ii. Diagramme de classe .....	23
b. Les collections rajoutées.....	24
1. Description des données.....	24
❖ users .....	24
❖ projects .....	24
❖ catalog.....	25
❖ Jointures.....	25
❖ usersDelete .....	26
❖ typeColumn .....	26
❖ Collections.....	27
2. Diagramme de classe .....	27
V. Création et administration de la base de données .....	29
1- Prérequis .....	29
Étape 1 .....	29
Étape 2 .....	29
Étape 3 .....	29
2- Ajouter un catalogue.....	29
Étape 1 .....	29
Description des champs .....	30
Étape 2 .....	30

➤ Exemple d'ajout de catalogue : le catalogue hs9.9.....	31
Étape 1 : Je modifie le fichier ajouter_catalogue_hs9.9.json .....	31
Étape 2 : J'exécute la fonction ajouter_un_projet :.....	32
Vérification.....	32
3- Ajouter un projet.....	32
Étape 1 .....	32
Étape 2 .....	32
Description des champs .....	33
Étape 3 .....	33
Exemple d'ajout de projet : le projet Microbaria .....	34
Étape 1 : Je modifie le fichier ajouter_un_projet.json.....	34
Étape 2 : J'exécute la fonction ajouter_un_projet :.....	35
Verification.....	35
4- Ajouter une collection.....	35
5- Ajouter des données via d'autre format que le Json.....	36
VI. Méthode d'optimisation des requêtes sur la base de données .....	37
VII. Description des variables principales.....	38
1. App .....	38
2. globalInformation .....	39
VIII. Présentation des modules et perspectives d'évolution.....	40
1. Les principaux modules.....	40
a. Connexion / login.....	40
Vu de l'interface .....	40
Description .....	40
Évolution .....	40
b. Accueil / Home.....	40
Vu de l'interface .....	40
Description .....	41
Évolution .....	41
c. Information connexion / connexion information .....	42
Vu de l'interface .....	42
Description .....	42

Évolution .....	42
d. Gestion des utilisateurs / users.....	43
Vu de l'interface.....	43
Description.....	43
Évolution .....	43
e. Gestion des projets / projects.....	44
Vu de l'interface.....	44
Description.....	44
Évolution .....	44
f. Collections.....	45
Vu de l'interface.....	45
Description.....	45
Évolution .....	45
g. Déconnexion / Log out.....	46
Vu de l'interface.....	46
Description.....	46
Évolution .....	46
2. Les modules fonctionnels .....	46
a. Générateur de requête / Make Queries .....	47
Vu de l'interface.....	47
Description.....	47
Évolution .....	47
Guide d'utilisation.....	47
Étape 1 : sélectionner, ordonner et valider .....	47
Étape 2 : personnaliser sa requête .....	49
Étape 3 : visualiser les résultats .....	51
Étape 4 : sauvegarder ma requête.....	52
b. Mes requêtes / My queries.....	53
Vu de l'interface.....	53
Description.....	54
Évolution .....	54
Guide d'utilisation.....	54

c.	Données des projets / Project data .....	57
	Vu de l'interface .....	57
	Description .....	57
	Évolution .....	57
	Guide d'utilisation .....	57
	Sélectionner des projets .....	58
	Analyser les abondances de MGS fusionnés avec les données des projets .....	58
	Comparer deux projets .....	60
d.	Sauvegarde /Save.....	62
	Vu de l'interface .....	62
	Description .....	62
	Évolution .....	62
	Guide d'utilisation .....	62
e.	Ma console / Console.....	63
	Vu de l'interface .....	63
	Description .....	63
	Évolution .....	63
	Guide d'utilisation .....	64
f.	Blast.....	64
	Vu de l'interface .....	64
	Description .....	65
	Évolution .....	65
	Guide d'utilisation .....	65
IX.	Ajout de nouveaux modules .....	67
	Étape 1 .....	67
	Étape 2 .....	67
	Étape supplémentaire .....	67
X.	Astuce et conseil .....	68
1.	Astuce.....	68
	Les noms de variables : .....	68
	Script volumineux .....	68
	Débogage du code : .....	68

2. Conseil.....	68
Les problèmes de développement :.....	68
Librairie existante :.....	68
XI. Perspective d'évolution .....	69
1. Niveau applicatif .....	69
2. Niveau structuration du code .....	69
3. Niveau interface.....	69
4. Niveau base de données.....	69
5. Niveau sauvegarde des données .....	69
6. Niveau Conception.....	70
7. Niveau fonctionnalité.....	70
XII. Mise en production de l'application .....	71
XIII. Liens utile .....	72
1. Documentations et astuce .....	72
Documentation mongolite.....	72
Débuter avec shiny.....	72
Débuter avec shinydashboard .....	72
ShinyAce .....	72
Documentation datatable .....	72
Démo blast .....	72
Icônes et couleur.....	73
Astuce pour développer avec Shiny.....	73
2. Aide à la Résolution de problème.....	73
Table des matières .....	76
Annexe .....	83
iogold database population .....	83
Introduction .....	83
The hs_9.9_annot_all object.....	83
The taxo_new object.....	86
Data extraction for MongoDB population .....	88
Export of tables for individual MGS .....	88
Export of MGS annotation table .....	88

Export of table for IGC genes with no MGS association .....	88
Summary .....	88
Microbaria iogold database population.....	89
Introduction .....	89
The gene abundance matrix .....	89
The MGS abundance matrix.....	89
The sample metadata matrix .....	90
Data extraction for MongoDB population .....	91
Export MGS abundance table .....	91
Export Sample metadata table .....	91
Export gene abundance table .....	91
Summary .....	92

## Annexe

### iogold database population

Eugenio Belda & Edi Prifti

30 May 2018

#### Introduction

The present document summarizes the data extraction process needed to populate iogold database based on IGC gene catalog (10 Million genes). This database will contain all functional annotation available for genes of the IGC catalog including stratification of genes in MGSs and taxonomic information of MGSs, and will integrate available IGC gene catalog annotations in Integromics servers (KO groups, COGs, and InterPro annotations) and updated annotations provided by the EMBL in the context of Metacardis project.

The prototype of iogold database will be created in MongoDB (to complete further details by Pape), and will be organized around MGSs (one data table for each MGSs containing all genes with the corresponding annotation) plus an additional data table that will contain all IGC genes without MGS assignment and a taxonomy data table with taxonomic annotation of MGSs.

The project will be structured around four main objectives:

1. Populate the prototype MongoDB with data tables of the different objects (MGSs, IGC genes without MGS assignment, MGS taxonomy)
2. Development of query functions for efficient access to information according with specific needs (p.ex. extract gene content table of MGSs based on a set of query functions)
3. Development of visualization interface (R. shiny)
4. Development of methods for extend the content of the database from additional sources of functional annotation (p.ex. Metacardis IGC catalog annotations)

The present document summarizes the data extraction process to populate iogold database based on IGC gene catalog (10 Million genes) needed to fulfill the first objective of the project.

#### The hs\_9.9\_annot\_all object

We will use this R data frame for the initial population of the MongoDB. This object contains the original annotation of the IGC catalog (9879896 lines/genes) plus additional annotations generated by InterproScan totalling 36 columns with different information about each IGC catalog gene:

x
GeneID
GeneName
GeneLength
GeneCompletenessStatus

CohortOrigin
TaxonomicPhylumAnnotation
TaxonomicGenusAnnotation
KOs
eggNOGs
SampleOccurrenceFrequency
IndividualOccurrenceFrequency
KEGGFunctionalCategories
eggNOGFunctionalCategories
CohortAssembled
interpro
GO
KEGG.MetaCyc.Reactome
Pfam
CDD
Coils
Gene3D
Hamap
Pirsf
PRINTS
Phobius
ProDom
PrositePatterns
PrositeProfiles
SMART
SUPERFAMILY
SignalP_GRAM_NEGATIVE
SignalP_GRAM_POSITIVE
TIGRFAM
TMHMM
KOs.embl.metacardis
MGS

As an example of the contents of this object for the first 6 genes and 9 first columns:

	GeneID	GeneName	GeneLength	GeneCompleteness	CohortOrigin	TaxonomicPhylumAnnotation	TaxonomicGenusAnnotation	KOs	eggNOGs
T2D-6A_GL0083352	1	T2D-6A_GL0083352	88230	Complete	CHN	unknown	unknown	K01824	COG5184
V1.UC4-5_GL0154511	2	V1.UC4-5_GL0154511	88086	Complete	EUR	unknown	unknown	K01824	COG5184
MH0198_GL0136826	3	MH0198_GL0136826	67464	Lack 5-end	EUR	unknown	unknown	K01824	COG5184

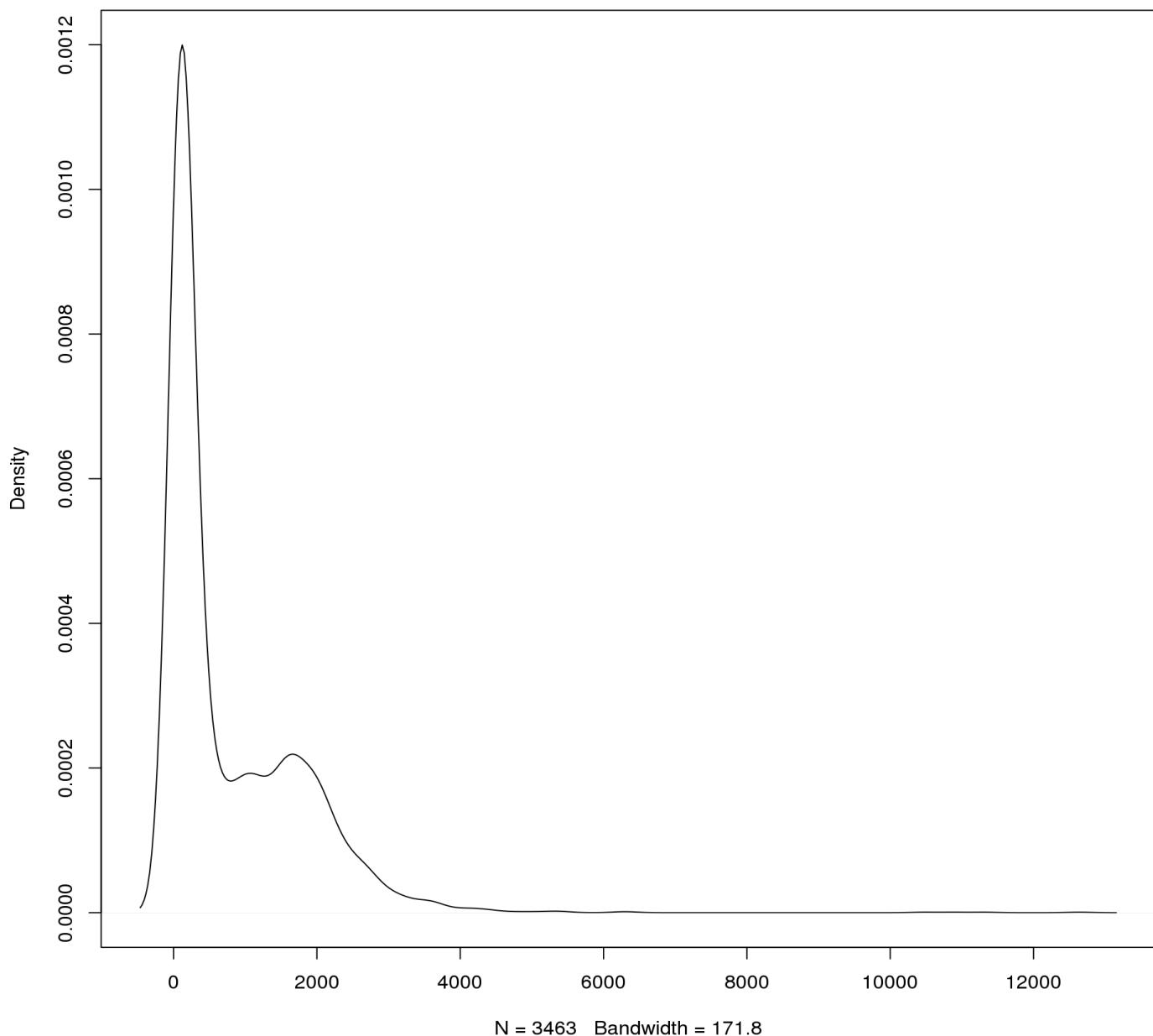
V1.FI28_GL0106390	4V1.FI28_GL0106390	45267	Complete	EUR	unknown	unknown	K03924	NOG12793
MH0318_GL0144550	5MH0318_GL0144550	42243	Complete	EUR	unknown	unknown	K13730	COG4886
SZEY-41A,GL0088289	6SZEY-41A,GL0088289	41277	Lack 5-end	CHN	unknown	unknown	K13735	COG5184

This object contains also information of the MGS assignment of different genes of the catalog in the last column:

	TMHMM	KOs.embl.metacardis	MGS
T2D-6A,GL0083352	unknown	K10615	NA
V1.UC4-5,GL0154511	unknown	K10595	NA
MH0198,GL0136826	unknown	K10595	NA
V1.FI28,GL0106390	TMhelix	K15125	CAG02058
MH0318,GL0144550	TMhelix	K15125	NA
SZEY-41A,GL0088289	unknown	K10615	NA

Overall, there are 3463 MGSs of a minimum size of 50 genes, comprising 2691408 catalog genes (27.24% of the total).

### MGS size distribution



7188488 catalog genes have no MGS assignment (72.26% of the total).

#### The `taxo_new` object

We will use this data frame as source of the taxonomic information about MGSs.

	size	NA_pc	BHit_pc	BHit_ali	BH_id	NA_pc	Taxo_poc	Taxo_poi	Taxo_ido	Taxo_leotvel	ann	species	genus	familiy	order	class	phylum	superkingdom

CA G00 001	126 38	78. 1	14. 5	Bla stoc ysti s ho mini s	65. 2	80. 5	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	
CA G00 002	113 13	90. 8	4.7	Bla stoc ysti s ho mini s	71. 4	82. 7	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	
CA G00 003	108 82	75. 8	16.	Bla 0stoc ysti s ho mini s	65. 7	80. 8	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	
CA G00 004	104 66	21. 6	77.	Bla 9stoc ysti s sp. ST4 (ob sole te)	98. 0	99. 7	21. 6	77.	Bla 9stoc ysti s sp. ST4 (ob sole te)	98. 0	99. 7	spe cies	Bla stoc ysti s sp. ST4 (ob sole te)	Bla stoc ysti s sp. ST4 (ob sole te)	Bla stoc ysti s sp. ST4 (ob sole te)	uncl assifi ed	uncl assifi ed	uncl assifi ed	uncl assifi ed	Euk aryo ota
CA G00 008	664 6	0.0	98.	[Clo 9strid ium] bolt eae 90A 5	99. 2	99. 1	0.0	100	Clo .0strid ium bolt eae	99. 7	99. 6	spe cies	Clo strid ium bolt eae	Clo strid ium bolt eae	Lac hno tridi um	Lac hno spir ace	Lac hno tridi um	Clo strid ia	Fir mic utesa	Bac
CA G00 009	633 3	14. 1	62.	Lac 0hno spir ace ae bact eriu m 3_1	91. 9	83. 5	14. 1	64.	uncl 0assi fied Lac hno spir ace ae	90. 8	83. 4	gen us	uncl assi fied Lac hno spir ace ae	NA	uncl assi fied Lac hno spir ace ae	Lac hno spir ace ae	Clo strid ia	Clo strid ia	Fir mic utesa	Bac

## Data extraction for MongoDB population

Based on discussions with Pape, the Mongodb will contain:

- One data table/object for each MGSs (3463 MGSs >50 genes=Minimum size of each data table)
  - One data table/object that will group all IGC genes without MGS assignment
  - One data table/object corresponding to MGS taxonomic annotations

For population of MongoDB, data tables should be in JSON format. We will save tables in /data/projects/iogold/data/microbiome/0.json.tables.ICGgenesMGS.MongoDB/.

## Export of tables for individual MGS

We will process hs\_9.9\_annot\_all data frame to extract individual tables for each MGS and save in JSON format ([MGSid].json files). We will use the R package rio for this export (<https://cran.r-project.org/web/packages/rio/vignettes/rio.html>)

## Export of MGS annotation table

Same procedure for saving MGS annotation table in json format (MGStaxonomy.json).

## Export of table for IGC genes with no MGS association

Same procedure but in this case we need to split and save the data into four chunks of 2 million genes each (not possible to save the 7188488 gene table in single json object=> *Error in collapse(tmp, inner = FALSE, indent = indent) : R character strings are limited to 2^31-1 bytes*).

Files saved for IGC genes with no MGS association have an extension NoMGSgenes[from,to].json.

## Summary

At the end, we have generated 3468 json files at

/data/projects/iogold/data/microbiome/0.json.tables.ICGgenesMGS.MongoDB:

- 3463 files corresponding to MGSs ([MGSid].json)
  - 4 files corresponding to the 7188488 IGC genes with no MGS association (NoMGSgenes[from.to].json)
  - 1 file (MGStaxonomy.json) with taxonomic information of MGSs

This files will be used for the initial population of the MongoDB.

# Microbaria iogold database population

Eugení Belda & Edi Prifti

06 September 2018

## Introduction

The present document summarizes the data extraction process needed to populate iogold database specific of Microbaria project. This represent a first test of a database specific of a research project. It will contain:

- Gene count matrix
- MGS abundance matrix
- Sample metadata

Abundance data will be based on the IGC gene catalog to fit with iogold reference database content.

## The gene abundance matrix

This matrix contains the normalized IGC gene abundance (downsized to 11M reads/sample) across 173 samples of Microbaria cohort:

- Rows corresponds to genes (7855540 genes). Rownames corresponds to GenID variable in IGC gene catalog annotation object used to populate iogold database (linking point to gene annotation data).
- Columns corresponds to sample IDs (173 samples). Column names will be the linking point to sample metadata

Example of the content of this gene count matrix

As an example of the contents of this object for the first 10 genes (rows) and 5 first samples (columns):

	MB16_3	MB50_1	MB08_1	MB37_3	MB29_1
1	0	0e+00	0e+00	0e+00	0e+00
2	0	0e+00	0e+00	0e+00	0e+00
3	0	0e+00	0e+00	0e+00	0e+00
4	0	0e+00	1e-07	0e+00	0e+00
6	0	3e-07	6e-07	1e-07	4e-07
7	0	0e+00	0e+00	0e+00	0e+00
9	0	0e+00	0e+00	0e+00	0e+00
10	0	0e+00	0e+00	0e+00	0e+00
11	0	0e+00	0e+00	0e+00	1e-07
12	0	0e+00	0e+00	0e+00	0e+00

## The MGS abundance matrix

This matrix contains the abundance data of 1436 metagenomic species (MGSs) of a minimum size of 500 genes across same 173 samples from microbaria cohort. MGS abundance is computed based on the mean abundances of the 50 MGS genes most correlated in terms of gene abundance (standard approach). In terms of structure:

- Rows corresponds to MGSs (1436 MGSs). Rownames corresponds to MGS IDs (CAGxxxxx

nomenclature). This will be the linking point to iogold reference database.

- Columns corresponds to sample IDs (173 samples). Column names will be the linking point to sample metadata

As an example of the contents of this object for the first 10 MGS (rows) and first 5 samples (columns):

	MB16_3	MB50_1	MB08_1	MB37_3	MB29_1
CAG00001	0.0e+00	0.00e+00	0	0.0e+00	0e+00
CAG00002	0.0e+00	0.00e+00	0	0.0e+00	0e+00
CAG00003	0.0e+00	0.00e+00	0	0.0e+00	0e+00
CAG00004	0.0e+00	0.00e+00	0	0.0e+00	0e+00
CAG00008	3.0e-07	0.00e+00	0	1.5e-06	1e-07
CAG00009	0.0e+00	0.00e+00	0	0.0e+00	0e+00
CAG00011	5.8e-06	1.64e-05	0	2.0e-07	0e+00
CAG00012	0.0e+00	0.00e+00	0	7.0e-07	0e+00
CAG00013	2.0e-07	0.00e+00	0	3.0e-07	0e+00
CAG00014	1.0e-07	0.00e+00	0	2.0e-07	0e+00

#### The sample metadata matrix

Microbaria project consist in the study of microbiome composition of severely obese patients before (T0 or baseline) and after (1, 3, 12 months) bariatric surgery (bypass or gastric banding). For further details see <https://www.ncbi.nlm.nih.gov/pubmed/?term=belda%20and%20prifti>.

We will use as sample metadata the gene richness matrix of the 173 microbaria samples based on IGC gene catalog plus information about patient, time and surgery type.

In terms of structure:

- Rows corresponds to sample IDs (173 samples). Row names will be the linking point to sample metadata
- Columns corresponds to 15 sample variables: \*\* Columns 1 to 12: Gene richness values of samples at different downsizing thresholds. \*\* Column 13: Patient ID (73 patients) \*\* Column 14: Time of sampling (T0, M1, M3, M12) \*\* Column 15: Surgery type (bypass/band)

As an example of the contents of this object for the first 10 samples (rows):

	down 3M una	down 5M una	down 10M una	down 11M una	down 15M una	down 20M una	down 25M una	down 30M una	down 35M una	down 40M una	down 45M una	down 50M una	Patie nt	Time	clin_s urg_t ype
MB0 1_3	2912 34	3628 28	4813 12	4995 27	5638 91	6242 83	6733 49	7151 78	7523 08	7851 25	8148 36	8408 53	micro baria 01	M3	bypa ss
MB0 1_1	3533 82	4335 08	5584 14	5770 88	6400 50	7014 38	7517 43	7945 92	8320 80	8679 40	9004 77	9292 18	micro baria 01	T0	bypa ss
MB0 1_2	2849 42	3503 73	4536 15	4691 88	5230 34	5766 09	6212 56	6597 97	6936 10	7243 39	7519 76	7759 93	micro baria 01	M1	bypa ss
MB0 2_2	3597 20	4498 38	5930 66	6149 92	6889 38	7618 63	8216 36	8725 29	9170 19	9570 54	9930 25	1026 059	micro baria 02	M1	bypa ss

MB0 2_1	3854 63	4786 27	6249 60	6469 22	7216 94	7951 21	8549 52	9063 39	9507 01	9902 30	1025 908	1059 059	micro baria 02	T0	bypa ss
MB0 4_2	2592 14	3242 41	4313 20	4478 52	5057 55	5601 34	6043 89	6424 20	6762 82	7061 98	7332 16	7566 36	micro baria 04	M1	bypa ss
MB0 4_3	4118 64	5166 04	6862 56	7124 19	8010 51	8904 09	9648 50	1022 735	1073 679	1118 756	1159 851	1196 844	micro baria 04	M3	bypa ss
MB0 4_1	3882 47	4805 42	6251 53	6467 66	7201 69	7922 12	8508 83	9009 29	9444 10	9832 45	1018 351	1050 584	micro baria 04	T0	bypa ss
MB0 4_4	5100 41	6324 85	8225 08	8504 05	9455 41	1038 513	1114 373	1178 161	1233 792	1283 326	1330 037	1372 445	micro baria 04	M12	bypa ss
MB0 5_2	4651 21	5673 26	7216 33	7444 41	8206 70	8952 68	9549 26	1006 197	1050 737	1090 131	1130 250	1166 302	micro baria 05	M1	bypa ss

#### Data extraction for MongoDB population

Based on discussions with Pape, we will convert these different datasets to json format in order to create the Mongodb of Microbaria project, which will contain:

- One data table/object for MGS abundance data
- One data table/object for gene abundance data
- One data table/object for sample metadata.

For population of MogoDB, data tables should be in JSON format. We will save tables in **/data/projects/iogold/data/microbiome/2.json.tables.Microbaria/**.

#### Export MGS abundance table

We will save MGS abundance table in JSON format (MicrobariaMGSabundance.json file). We will use the R package rio for this export (<https://cran.r-project.org/web/packages/rio/vignettes/rio.html>)

#### Export Sample metadata table

We will save sample metadata table in JSON format (MicrobariaSampleMetadata.json file).

#### Export gene abundance table

Same procedure for saving gene abundance table in json format (MicrobariaGeneAbundance.json). In this case we need to split and save the data into 8 chunks of 7 million genes each (not possible to save the entire gene abundance table in single json object=> *Error in collapse(tmp, inner = FALSE, indent = indent) : R character strings are limited to 2^31-1 bytes*).

## Summary

At the end, we have generated 6 json files at /data/projects/iogold/data/microbiome/2.json.tables.Microbaria:

- 1 file corresponding to MGS abundance data (MicrobariaMGSabundance.json)
- 1 file corresponding to sample metadata (MicrobariaSampleMetadata.json)
- 8 files corresponding to splitted gene abundance matrix

This files will be used for the initial population of the Microbaria project MongoDB.