# Documentation technique

## Introduction :

Ce document technique fournit une explication technique de notre application Shiny « Vélov à Lyon ». Le code est conçu pour créer une application interactive permettant aux utilisateurs d'explorer les statistiques en temps réel des stations Vélov à Lyon

## Initialisation :

Le début du script est consacré à l'installation des packages nécessaires, tels que httr, jsonlite, shiny, ggplot2, plotly, cli, leaflet et shinythemes. Ces packages sont essentiels pour la manipulation de données, la création d'interfaces Shiny, la création de la carte et la génération de graphiques interactifs.

## Appel à l'API JCDecaux :

Après l'initialisation des packages, le code fait appel à l'API JCDecaux pour récupérer en temps réel les informations sur les stations Vélov à Lyon. Les paramètres du contrat et la clé API sont définis, et une requête HTTP est effectuée via la fonction GET de httr. Les données JSON obtenues sont converties en objets R avec fromJSON.

## Fichier « carte.csv » :

On créer le fichier contenant les codes postaux, qui nous servira pour filtrer les stations sur notre carte avec le code postal csv grâce à ses quelques lignes de code :

#création fichier csv

data$lattitude<-data$position.latitude

data$longitude<-data$position.longitude

# géocodage

install.packages('tidygeocoder')

library(dplyr, warn.conflicts = FALSE)

library(tidygeocoder)

geo<-reverse\_geocode(data,lat = lattitude , long = longitude , method = 'osm',

address = address\_found, full\_results = TRUE)

#création d'un nouveau data frame pour les position geo et les code postaux

carte<-data.frame(number=data$number,codepostale=geo$postcode)

write.table(carte,"L:/BUT/SD/Promo 2022/lchaurand/Rshiny porjet/carte.csv",row.names=F)

Puis on l’ouvre. Pour cela on utilise la fonction read.csv().

## Création d’une base SQL :

On créer une base sur le site free sql data base en faisant un lien entre notre base R et la base SQL. Il suffit d’exécuter les quelques lignes de codes suivantes :

#création base

install.packages("RMySQL")

library(RMySQL)

#créer connexion entre base R et abse SQL

con<-dbConnect(MySQL(),

user="sql11646646",

password="2zLzyEfkSi",

host="sql11.freesqldatabase.com",

dbname="sql11646646")

summary(con)

dbGetInfo(con)

#créer table

dbWriteTable(con,"tablere",data)

#connaitre les tables de la BDD

dbListTables(con)

## L’interface utilisateur :

* Il représente la structure générale de l'interface Shiny avec quatre onglets.
* Chaque onglet a sa propre mise en page avec des éléments de barre latérale (sidebar) et de panneau principal (main).
* Il contient aussi des widgets, Certains comme les cases à cocher checkboxGroupInput(), des sliders sliderInput() ou des menus déroulants selectInput() permettent à l’utilisateur de choisir comment filtrer les graphiques ou les sorties image ou texte.
* Il contient aussi les graphiques dont on décide l’emplacement dans nos différents onglets avec plotlyOutput().

## Le server :

Le serveur contient le code pour réagir aux actions de l'utilisateur et générer des sorties dynamiques. Il est organisé en fonction des onglets de l'interface.

#### Accueil :

* Un bouton "Rafraichir les données" permet de mettre à jour les données en appelant à nouveau l'API JCDecaux.
* Une jointure est faite pour faire la jointure entre le fichier csv « carte.csv », contenant les numéros des stations ainsi que les codes postaux associés. On fait cela grâce à la fonction merge().
* On créer ensuite une carte qui affichera les stations Vélov à Lyon avec leur nom et le nombre de vélos disponible grâce à la fonction leaflet().

#### Vélos disponibles :

* Affiche le nombre de vélos disponibles par station dans un graphique interactif grâce à plotly().
* Un message informe l'utilisateur si aucune station n'est sélectionnée ou si aucun vélo n'est disponible.

#### Places disponibles :

* Présente deux graphiques interactifs. Le premier montrant les stations qui disposent du nombre de places sélectionnées par l’utilisateur. Le second montre le nombre de places disponibles par station sélectionnées aussi par l’utilisateur.
* Des messages avertissent si aucune station correspondante n'est trouvée ou si le nombre de places sélectionnées est nul.

#### Type de vélos :

* Le premier onglet permet à l'utilisateur de sélectionner un type de vélo (mécanique ou électrique) et affiche des informations associées, une image et une description.
* Le deuxième onglet présente la répartition du nombre de vélos par type dans une station sélectionnée par l’utilisateur.