Application for the Database

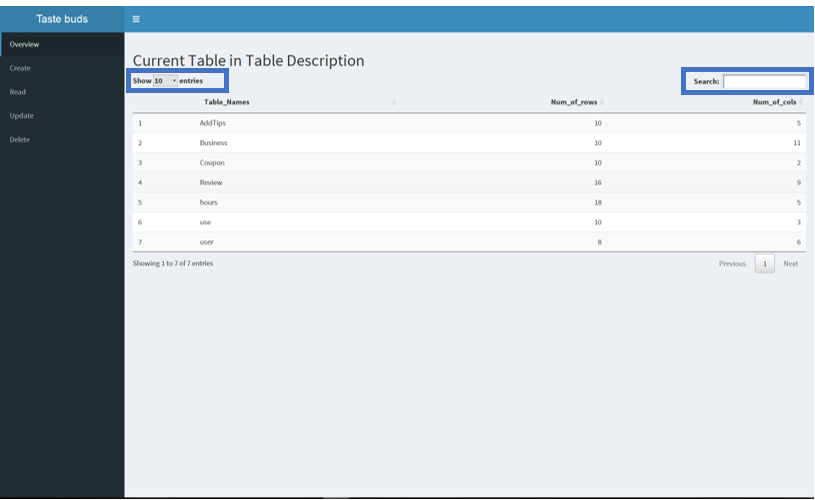
1. **Introduction**

This user-friendly website page is an application created in R, linking to the database, and its interface could well realize the function of overviewing, filtering and altering the data.

The web address is <http://127.0.0.1:7836>.

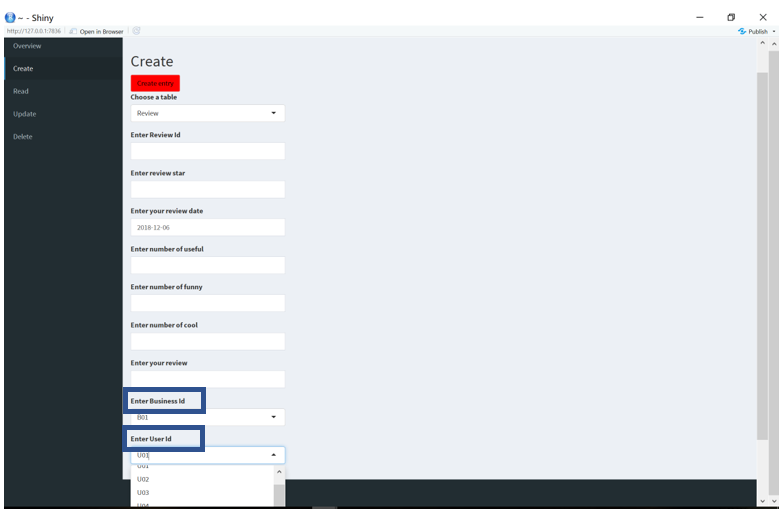
1. **Function**
2. Overview

In the Overview part, the customer/ business owner could check the overall status of the database, including the total number of the attributes and records in each table. They can also manage the records shown with the select box on the left, and type in the key word in the right text box on the right to search specific data they want.



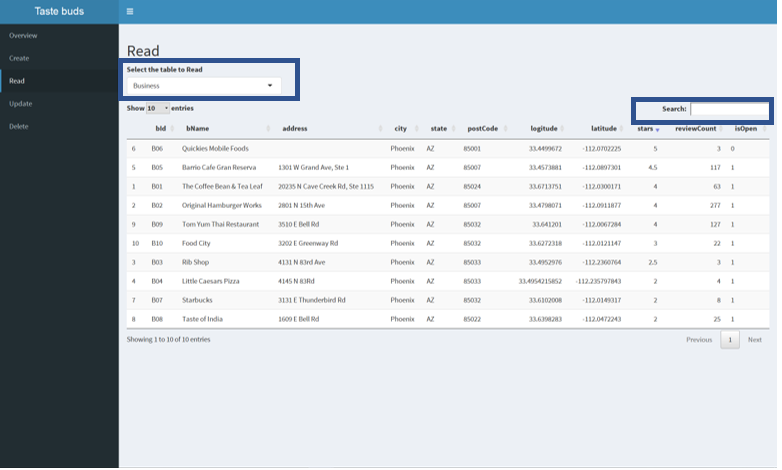
1. Create

The creating part is exclusively for the customers to add new records into the Review table, so the business owners will not have access to this part. The page contains multiple text box to make it convenient for the customer to leave their reviews by attributes and also to maintain the integrity of the database itself. Since the bId and uId together composite the primary key in the Review Table, these two cannot be null, so we create with the select box instead of the text box.



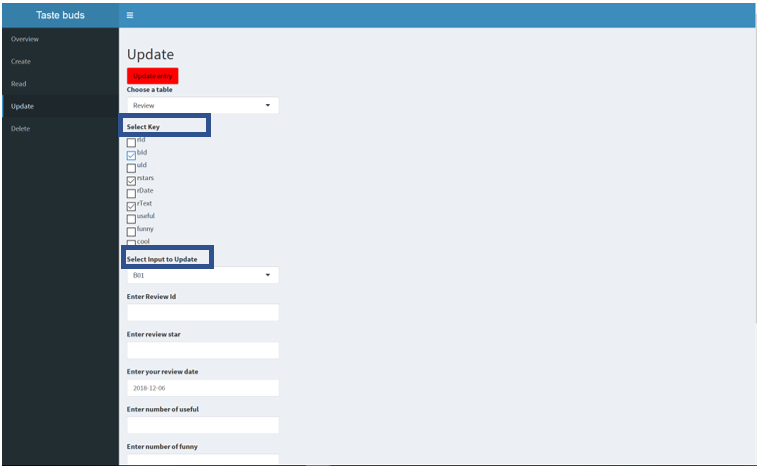
1. Read

Unlike the overview part, the Read function is for the customer/ business owner accessing the detail data in each data. From the select box, they can choose which table to view, and they can also manage the records shown with the select box on the left, and type in the key word in the right text box on the right to track one specific record.



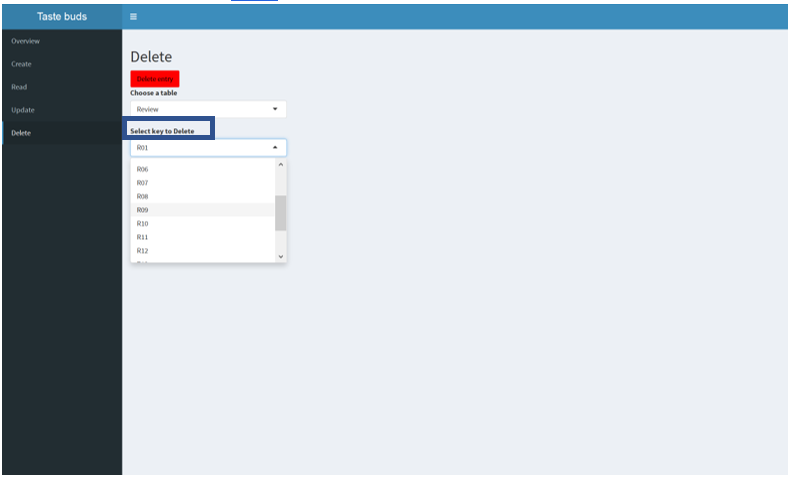
1. Update

This part is for the customers to alter/ change/ update their former records in the Review table, still the business owners will not have access to it. For precisely locating the review once created, the customer could first select anyone of the criteria about the business that they can recall from the list of check boxes, and after that they can select from all the options under that criterion in the following select box. Then they can type in whichever text boxes they want to change and click update to alter the data.



1. Delete

In this part, the customer could delete any review that they are not satisfied with, and they can merely select the Review ID of the one they want to delete and click delete button.



1. **Appendix: code in R**

library(shiny)

library(shinydashboard)

library(dplyr)

library(tibble)

library(pool)

library(rlang)

library(DT)

pool <- dbPool(RSQLite::SQLite(), dbname = "butan.db")

allTables <- db\_list\_tables(pool)

ui <- dashboardPage(

dashboardHeader(title = "Taste buds"),

dashboardSidebar(

sidebarMenu(id = "tabs",

menuItem("Overview", tabName = "dashboard"),

menuItem("Create", tabName = "create"),

menuItem("Read", tabName = "read"),

menuItem("Update", tabName = "update"),

menuItem("Delete", tabName = "delete")

)

),

dashboardBody(

tabItems(

tabItem(tabName = "dashboard",

h2("Current Table in Table Description"),

dataTableOutput(outputId = "overview")

),

tabItem(tabName = "create",

h2("Create"),

actionButton(

inputId = "Insert",

label = "Create entry",

style="color: black; background-color: red"

),

selectInput(

inputId = "create\_table",

label = "Choose a table",

choices = 'Review'

),

uiOutput(outputId = "fields")

),

tabItem(tabName = "read",

h2("Read"),

selectInput(

inputId = "select\_table",

label = "Select the table to Read",

choices =allTables

),

dataTableOutput(outputId = "read\_table")

),

tabItem(tabName = "update",

h2("Update"),

actionButton(

inputId = "Update",

label = "Update entry",

style="color: black; background-color: red"

),

selectInput(

inputId = "update\_table",

label = "Choose a table",

choices = 'Review'

),

checkboxGroupInput(inputId = "key", label = "Select Key",

choices = NULL),

selectInput(

inputId = "rId",

label = "Select Input to Update",

choices = NULL

),

uiOutput(outputId = "fields\_u")

),

tabItem(tabName = "delete",

h2("Delete"),

actionButton(

inputId = "Delete",

label = "Delete entry",

style="color: black; background-color: red"

),

selectInput(

inputId = "delete\_table",

label = "Choose a table",

choices = 'Review'

),

selectInput(

inputId = "key\_id",

label = "Select key to Delete",

choices = NULL

)

)

)

)

)

server <- function(input, output, session) {

output$overview <- renderDataTable({

allTables <- as.data.frame(allTables)

num\_of\_col <- c()

num\_of\_row <- c()

for (i in 1: nrow(allTables)){

num\_of\_col <- c(num\_of\_col, length(db\_query\_fields(pool, allTables[ i , 1])))

num\_of\_row <- c(num\_of\_row, db\_query\_rows(pool, allTables[ i , 1]))

}

allTables <- cbind(allTables, num\_of\_row, num\_of\_col)

names(allTables) <- c("Table\_Names", "Num\_of\_rows", "Num\_of\_cols")

datatable(allTables)

})

output$read\_table <- renderDataTable({

x <- data.frame(pool %>% tbl(input$select\_table))

datatable(x)

})

output$fields <- renderUI({

if (sym(input$create\_table) == "Review"){

Business <- as.data.frame(pool %>% tbl("Business"))

user <- as.data.frame(pool %>% tbl("user"))

tagList(

textInput("rId", "Enter Review Id"),

textInput("rstars", "Enter review star"),

dateInput("rDate", "Enter your review date"),

textInput("useful", "Enter number of useful"),

textInput("funny", "Enter number of funny"),

textInput("cool", "Enter number of cool"),

textInput("rText", "Enter your review"),

selectInput("bId", label = "Enter Business Id", choices = Business$bId),

selectInput("uId", label = "Enter User Id", choices = user$uId)

)

}

})

observeEvent(input$Insert, {

df <- data.frame(

rId = input$rId,bId = input$bId,uId = input$uId,

rstars = input$rstars,rDate = input$rDate,rtext = input$rtext,

useful = input$useful,funny = input$funny,cool = input$cool,

stringsAsFactors = FALSE)

print(df)

sql <- "INSERT INTO ?table VALUES(?rId, ?bId, ?uId,

?rstars, ?rDate, ?rText, ?useful, ?funny, ?cool);"

query <- sqlInterpolate(pool, sql, .dots = c(

list(table = input$create\_table),

as\_list(df)

))

print(query)

dbExecute(pool, query)

})

observe({

req(input$update\_table)

cols <- db\_query\_fields(pool, input$update\_table)

updateCheckboxGroupInput(session, "key", choices = cols)

})

observe({

tbl <- data.frame(pool %>% tbl(input$update\_table) )

updateSelectInput(session, "rId", choices = tbl[,input$key] )

})

output$fields\_u <- renderUI({

req(input$rId)

if (sym(input$create\_table) == "Review"){

Business <- as.data.frame(pool %>% tbl("Business"))

user <- as.data.frame(pool %>% tbl("user"))

tagList(

textInput("rId", "Enter Review Id"),

textInput("rstars", "Enter review star"),

dateInput("rDate", "Enter your review date"),

textInput("useful", "Enter number of useful"),

textInput("funny", "Enter number of funny"),

textInput("cool", "Enter number of cool"),

textInput("rText", "Enter your review"),

selectInput("bId", label = "Enter Business Id", choices = Business$bId),

selectInput("uId", label = "Enter User Id", choices = user$uId)

)

}

})

observeEvent(input$Insert, {

df <- data.frame(

rId = input$rId,bId = input$bId,uId = input$uId,

rstars = input$rstars,rDate = input$rDate,rtext = input$rtext,

useful = input$useful,funny = input$funny,cool = input$cool,

stringsAsFactors = FALSE)

sql <- "UPDATE ?table SET bId = ?bId, uId = ?uId,

rstars = ?rstars, rDate = ?rDate, rtext = ?rtext, useful = ?useful,

funny = ?funny, cool = ?cool WHERE rId = ?idVal;"

query <- sqlInterpolate(pool, sql, .dots = c(

list(table = input$update\_table),

as\_list(df),

list(idVal = input$rId)

))

dbExecute(pool, query)

})

observe({

if(sym(input$delete\_table) == "Review"){

tbl <- data.frame(pool %>% tbl(input$delete\_table) )

print(tbl)

updateSelectInput(session, "key\_id", choices = tbl$rId )

}

})

observeEvent(input$Delete, {

sql <- "DELETE FROM ?table WHERE rId = ?idVal;"

query <- sqlInterpolate(pool, sql, .dots = c(

list(table = input$delete\_table),

list(idVal = input$key\_id)

))

dbExecute(pool, query)

})

}

shinyApp(ui, server)