Coding Report Outline

Due April 17, 2020

Team 3- Gracie, Katie, Ben, Abby, Brad, Owen, Ana

1. List of files
    - app.R: runs on PC when deployed locally for development, will run on the Shiny IO server when deployed for users (slip into across two different files for this report)
    - utils.R: runs on PC when deployed locally for development, will run on the Shiny IO server when deployed for users
    - libraries.R: runs on PC when deployed locally for development, will run on the Shiny IO server when deployed for users
    - dataIntake.R: runs on PC when deployed locally for development, will run on the Shiny IO server when deployed for users
    - grade_book_data.xlsx: not a code file, but created this Excel sheet of stub data to fill our database with (not included in the report)

```r
# Project: Mastery Grade System
# Professor Side Dashboard
# app.R
#
# 4/16/20 -- First release (MIT License), in class demo
#
# Purpose:This file contains the professor side server and user interface. This is where the main development of our app takes
place.
#
# Authors: Owen Bezick, Ana Hayne, Katie Turner, Brad Shook, Abby Santiago,
# Ben Santiago, and Gracie Petty
#

# Source Libraries
source("libraries.R", local = TRUE)
source("dataIntake.R", local = TRUE)

# UI ----
ui <- dashboardPage(
  dashboardHeader(title = "Professor View"
  )
  # Sidebar ----
  , dashboardSidebar(
    sidebarMenu(
      menuItem(tabName = "home", text = "Home", icon = icon("home"))
      , menuItem(tabName ="viewGrades", text = "View Grades", icon = icon("chalkboard")
                 , menuSubItem(tabName = "reviewGrades", text = "View Review Grades")
                 , menuSubItem(tabName = "homeworkGrades", text = "View Homework Grades")
      )
      , menuItem(tabName ="editGrades", text = "Edit Grades", icon = icon("chalkboard-teacher")
                 , menuSubItem(tabName = "editReviewGrades", text = "Edit Review Grades")
                 , menuSubItem(tabName = "editHomeworkGrades", text = "Edit Homework Grades")
      )
    )
  )
  # Body ----
  , dashboardBody( # Contains tabItems
    tabItems(
      # Home UI ----
      tabItem(
        tabName = "home"
        , HTML("<center><h1> Mastery Gradebook Dashboard </h1></center>")
        , div(img(src="davidsonCollege.jpg"), style="text-align: center;")
        , HTML("<center> <h3> Software Design, Group 3. <br> Gracie Petty, Abby Santiago, Ben Santiago, Brad Shook,
Katie Turner, Ana Hayne & Owen Bezick </h3></center>")
      )
      # View Review UI ----
      , tabItem(
        tabName = "reviewGrades"
        ,fluidRow(
          box(width = 12, title = "Filter:", status = "primary"
            ,column(width = 6
                    ,uiOutput("reviewStudentPicker")
            )
            , column(width = 6
                     ,uiOutput("reviewPicker")
            )
          )
        )
        , fluidRow(
          box(width = 6, status = "primary", title = "Review Grades", height = "550"
            , DTOutput("totalReviewGrades")
          )
          , box(width = 6, height = "550", stauts = "primary", title = "Total Grades", status = "primary"
                , echarts4rOutput("gradeBar"))
        )
      )
      # View Homework UI ----
      , tabItem(
        tabName = "homeworkGrades"
        , fluidRow(
          box(width = 12, title = "Filter:", status = "primary"
            ,column(width = 6
                , uiOutput("hwStudentPicker")
            )
            , column(width = 6
                ,uiOutput("hwPicker")
            )
          )
        )
        , fluidRow(
          box(width = 6, status = "primary", height= "550", title = "Homework Grades"
            , DTOutput("homeworkGradeTable")
          )
          , box(width = 6, status = "primary", height= "550", title = "Homework Averages"
             , echarts4rOutput("avgHomeworkGraph")
          )
        )
      )
      # Edit Review Grades ----
      , tabItem(
        tabName = "editReviewGrades"
        , fluidRow(
          box(width = 12, status = "primary", title = "Edit Review Grades"
            , column(width = 12
                , DTOutput("totalEditReviewGrades")
            )
          )
        )
      )
      # Edit Homework UI ----
      , tabItem(
        tabName = "editHomeworkGrades"
        , fluidRow(
          box(width = 12, status = "primary", title = "Edit Homework Grades"
            , DTOutput("editHomeworkGrades")
          )
        )
      )

    )
  )
)


# Define server logic
server <- function(input, output) {
  # View Review Server ----
  # List of students by ID
  ls_studentsR <- reactive({
    df <- getReviewGrades()
    df %>% distinct(firstLast) %>% pull()
  })
  # List of students by first_name
  #List of reviews
  ls_reviews <- reactive({
    df <- getReviewGrades()
    df %>% distinct(review_id) %>% pull()
  })
  # Student Picker
  output$reviewStudentPicker <- renderUI({
    pickerInput("reviewStudentPicker"
          ,"Student"
```

```
              , choices = ls_studentsR()
              , selected = ls_studentsR()
              , multiple = TRUE)
})
# Review Picker
output$reviewPicker <- renderUI({
  pickerInput("reviewPicker"
              ,"Review by ID"
              , choices = ls_reviews()
              , selected = ls_reviews()
              , multiple = TRUE)
})

# DT output
output$totalReviewGrades <- renderDT({
  req(input$reviewStudentPicker, input$reviewPicker)
  df <- getReviewGrades()
  df <- df %>%
    filter(review_id %in% input$reviewPicker, firstLast %in% input$reviewStudentPicker) %>%
    select( First = first_name, Last = last_name,`Review ID` = review_id, Topic = topic_id, Grade = grade)
  datatable(df, rownames = FALSE)
})

# Total Grades Chart
output$gradeBar <- renderEcharts4r({
  req(input$reviewStudentPicker, input$reviewPicker)
  df <- getReviewGrades() %>%
    filter(review_id %in% input$reviewPicker, firstLast %in% input$reviewStudentPicker) %>%
    select(grade) %>%
    count(grade)
  graph_df <- as_data_frame(t(df)) %>%
    mutate(chart = "")

  graph_df %>%
    e_chart(chart) %>%
    e_bar("V1", name = "Apprentice") %>%
    e_bar("V2", name = "Journeyman")  %>%
    e_bar("V3", name = "Master") %>%
    e_theme("westeros") %>%
    e_tooltip() %>%
    e_legend(bottom = 0)
})

# View Homeworks Server -----
# List of students by firstLast
ls_studentsHW <- reactive({
  df <- getHomeworkGrades()
  df %>% distinct(firstLast) %>% pull()
})
#List from homework
ls_homeworksHW <- reactive({
  df <- getHomeworkGrades()
  df %>% distinct(homework_id) %>% pull()
})
# Student Picker
output$hwStudentPicker <- renderUI({
  pickerInput("hwStudentPicker"
              ,"Student"
              , choices = ls_studentsHW()
              , selected = ls_studentsHW()
              , multiple = TRUE)
})
# Homework Picker
output$hwPicker <- renderUI({
  pickerInput("hwPicker"
              ,"Homework by ID"
              , choices = ls_homeworksHW()
```

```r
                , selected = ls_homeworksHW()
                , multiple = TRUE)
  })
  # Table
  output$homeworkGradeTable <- renderDT({
    req(input$hwStudentPicker, input$hwPicker)
    df <- getHomeworkGrades()
    df  <- df %>%
      filter(firstLast %in% input$hwStudentPicker) %>%
      filter(homework_id %in% input$hwPicker) %>%
      select(First = first_name, Last = last_name, `Homework ID` = homework_id, Grade= grade)

    datatable(df, rownames = FALSE)
  })

  # Homework Average Graph
  # Data
  hwAvg <- reactive({
    req(input$hwStudentPicker, input$hwPicker)
    df <- getHomeworkGrades()
    df  <- df %>%
      filter(firstLast %in% input$hwStudentPicker) %>%
      filter(homework_id %in% input$hwPicker) %>%
      group_by(student_id) %>%
      mutate(homeworkAvg = mean(grade)/100)
  })
  # Graph
  output$avgHomeworkGraph <- renderEcharts4r({
    df <- hwAvg()
    df %>%
      e_chart(last_name) %>%
      e_scatter(homeworkAvg, symbol_size = 10) %>%
      e_theme("westeros") %>%
      e_tooltip(formatter = e_tooltip_item_formatter(
        style = c("percent"),
        digits = 2
      )
      ) %>%
      e_x_axis(axisLabel = list(interval = 0, rotate = 45)) %>%
      e_y_axis(formatter = e_axis_formatter(
        style = c("percent"),
        digits = 2,
      )
      ) %>%
      e_legend(show = F)
  })
  # Edit Review Server ----
  # DT output
  reviewGradesData <- reactive({
    df <- getReviewGrades()
    df <- df %>%
      select(First = first_name, Last = last_name,`Review Id` = review_id,Topic = topic_id, Grade = grade)
  })

  output$totalEditReviewGrades <- renderDT({
    df <- reviewGradesData()
    datatable(df, rownames = FALSE, selection = list(mode = 'single', target = 'row'), filter = 'top', caption = "Click a Row to
Edit")
  })

  observeEvent(input$totalEditReviewGrades_rows_selected,{
    rowNumber <- input$totalEditReviewGrades_rows_selected
    df <- reviewGradesData()
    rowData <- df[rowNumber, ]
    showModal(
      modalDialog(title = "Edit Grade", easyClose = T
                  ,box(width = 12, status = "primary"
                        , HTML("<b> Name: </b>")
                        , renderText(paste(rowData$First, rowData$Last))
                        , HTML("<b> Topic ID: </b>")
                        , renderText(rowData$Topic)
                        , pickerInput("grade", "Grade:", choices = c("M", "J", "A"), selected = as.character(rowData$Grade))
                    )
                    , footer = fluidRow(
                        column(width = 6
                              , actionBttn("gradeSave"
                                          , "Save"
                                          , icon = icon("save")
                                          , style = "material-flat"
                                          , block = T
                              )
                        )
                        , column(width = 6
                                , actionBttn("gradeDismiss"
                                            , "Dismiss"
                                            , icon = icon("close")
                                            , style = "material-flat"
                                            , block = T)
                        )
                    )
            )
        )
    })

    # When the "Grade Dismiss" button is pressed
    observeEvent(input$gradeDismiss,{
        removeModal()
    })

    #When the "Save Grade" button is pressed
    observeEvent(input$gradeSave,{
        rowNumber <- input$totalEditReviewGrades_rows_selected
        df <- reviewGradesData()
        rowData <- df[rowNumber, ]
        topic_id <- rowData$Topic
        newGrade <- as.character(input$grade)
        review_id <- rowData[1, 3]

        df <- df_students %>%
            filter(first_name == rowData$First) %>%
            filter(last_name == rowData$Last)

        student_id <- df$student_id

        # Write to Database
        sql_query <- paste0("update Shiny.dbo.reviewGrades set grade = '", newGrade, "' where (topic_id = ", topic_id, " and
student_id = ", student_id, " and review_id = ", review_id, ")")
        dbExecute(con, sql_query)

        # Background App Refresh
        sql_query <- 'Select * from Shiny.dbo.reviewGrades'
        df_reviewGrades <- dbGetQuery(con, sql_query)
        reactive$df_reviewGrades <- df_reviewGrades

        showNotification("Changes Saved to Remote Database.", type = c("message"), duration = 3)

        removeModal()
    })

    # Edit Homework Server ----
    # DT output
    homeworkGradesData <- reactive({
        df <- getHomeworkGrades()
        df <- df %>%
      select(First = first_name, Last = last_name,`Homework Id` = homework_id, Grade = grade)
  })

  output$editHomeworkGrades <- renderDT({
    df <- homeworkGradesData()
    datatable(df, rownames = FALSE, selection = list(mode = 'single', target = 'row'), filter = 'top', caption = "Click a Row to
Edit")
  })

  observeEvent(input$editHomeworkGrades_rows_selected,{
    rowNumber <- input$editHomeworkGrades_rows_selected
    df <- homeworkGradesData()
    rowData <- df[rowNumber, ]
    showModal(
      modalDialog(title = "Edit Grade", easyClose = T
            ,box(width = 12, status = "primary"
               , HTML("<b> Name: </b>")
               , renderText(paste(rowData$First, rowData$Last))
               , HTML("<b> Homework ID: </b>")
               , renderText(rowData[1,3])
               , numericInput("hwGrade", "Grade:",  value = as.numeric(rowData$Grade))
            )
            , footer = fluidRow(
                column(width = 6
                     , actionBttn("hwgradeSave"
                             , "Save"
                             , icon = icon("save")
                             , style = "material-flat"
                             , block = T
                     )
                )
                , column(width = 6
                      , actionBttn("hwgradeDismiss"
                              , "Dismiss"
                              , icon = icon("close")
                              , style = "material-flat"
                              , block = T)
                )
            )
      )
    )
  })

  observeEvent(input$hwgradeDismiss,{
    removeModal()
  })
  observeEvent(input$hwgradeSave,{
    # Write to Database and pull
    removeModal()
  })

}

# Run the application
shinyApp(ui = ui, server = server)
```

```r
# Project: Mastery Grade System
# Professor Side Dashboard
# utils.R
#
# 4/16/20 -- First release (MIT License), in class demo
#
# Purpose:This file connects the application to a SQL database
#
# Authors: Owen Bezick, Ana Hayne, Katie Turner, Brad Shook, Abby Santiago,
# Ben Santiago, and Gracie Petty
#

# Create SQL table from R dataframe
tbl_create <- function(con, data, name) {
  copy_to(
    dest = con,
    df = data,
    name = name,
    overwrite = TRUE,
    temporary = FALSE
  )
}

# Connect to SQL database
db_connect <- function(
  server = "mydbinstance.c0eoxulijuju.us-east-2.rds.amazonaws.com",
  database = "shiny",
  uid = "datacats",
  pwd = "davidson",
  port = 1433,
  tds_version = 9.0,
  local = Sys.getenv('SHINY_PORT') == ""
) {
  if (local) {
    dbConnect(
      odbc(),
      Driver = "ODBC Driver 17 for SQL Server",
      Server = server,
      Database = database,
      uid = uid,
      pwd = pwd
    )
  } else {
    dbConnect(
      odbc(),
      Driver   = "libtdsodbc.so",
      Database = database,
      Uid      = uid,
      Pwd      = pwd,
      Server   = server,
      Port     = port
    )
  }
}

# Used when appending dataframes to correct quotes for SQL
quotes <- function(df) {
  for (c in 1:ncol(df))
    if (!class(df[,c]) %in% c("numeric", "integer")){
      df[,c] <- sQuote(df[,c], options(useFancyQuotes = FALSE))
    }
  df
}
```

```r
# Project: Mastery Grade System
# Professor Side Dashboard
# dataIntake.R
#
# 4/16/20 -- First release (MIT License), in class demo
#
# Purpose:This file loads stub data from an excel sheet into our SQL database
#
# Authors: Owen Bezick, Ana Hayne, Katie Turner, Brad Shook, Abby Santiago,
# Ben Santiago, and Gracie Petty
#


# DB Connnection ----
con  <- db_connect()

# Pull each data table from SQL ----
sql_query <- 'Select * from Shiny.dbo.topics'
df_topics <- dbGetQuery(con, sql_query)

sql_query <- 'Select * from Shiny.dbo.reviews'
df_reviews <- dbGetQuery(con, sql_query)

sql_query <- 'Select * from Shiny.dbo.homeworks'
df_homeworks <- dbGetQuery(con, sql_query)

sql_query <- 'Select * from Shiny.dbo.students'
df_students <- dbGetQuery(con, sql_query)

sql_query <- 'Select * from Shiny.dbo.reviewGrades'
df_reviewGrades <- dbGetQuery(con, sql_query)

sql_query <- 'Select * from Shiny.dbo.homeworkGrades'
df_homeworkGrades <- dbGetQuery(con, sql_query)

reactive <- reactiveValues(df_reviewGrades = df_reviewGrades)

# Get Homework Grades
getHomeworkGrades <- reactive({
  merge(df_homeworkGrades, df_students) %>% merge(df_homeworks) %>%
    mutate(firstLast = paste(first_name, last_name))
})

# Get Review Grades
getReviewGrades <- reactive({
  merge(reactive$df_reviewGrades, df_students) %>% merge(df_reviews) %>%
    mutate(firstLast = paste(first_name, last_name))
})
```

```r
# Project: Mastery Grade System
# Professor Side Dashboard
# libraries.R
# 4/16/20 -- First release (MIT License), in class demo
#
# Purpose:This file includes all of the necessary libraries for our application
#
# Authors: Owen Bezick, Ana Hayne, Katie Turner, Brad Shook, Abby Santiago,
# Ben Santiago, and Gracie Petty
#

# Libraries

# Shiny
library(shiny)
library(shinydashboard)
library(shinydashboardPlus)
library(shinyWidgets)

# Data
library(DT)
library(dplyr)
library(lubridate)
library(openxlsx)
library(tidyverse)
library(readxl)

# Viz
library(echarts4r)

# SQL Drivers
library(odbc)
library(DBI)
```