STAT 29000 Project 9

Topics: R, Shiny

Motivation: For what you can create using RShiny, it has a very low barrier to entry. You can throw together simple apps or dashboards quickly. RShiny is also a great tool for interactive demonstrations or presentations. In this project we will assemble an app that provides users with an interactive demonstration of how usernames and passwords work at a high-level.

Context: We've jumped right into building shiny apps. We've learned about the different parts of a shiny app, how to control the layout, etc. In this project, we will continue to practice what we learned before, and simultaneously learning about a tangentially related topic – authentication using usernames and passwords.

Scope: Using the `shiny` package to build simple web apps.

Rstudio provides *excellent* examples that are instantly available for you to play around with:

```
library(shiny)
runExample("01_hello")      # a histogram
runExample("02_text")       # tables and data frames
runExample("03_reactivity") # a reactive expression
runExample("04_mpg")        # global variables
runExample("05_sliders")    # slider bars
runExample("06_tabsets")    # tabbed panels
runExample("07_widgets")    # help text and submit buttons
runExample("08_html")       # Shiny app built from HTML
runExample("09_upload")     # file upload wizard
runExample("10_download")   # file download wizard
runExample("11_timer")      # an automated timer
```

By running one of the examples you are immediately presented with an app for you to test out, as well as the associated (copy-and-pastable) code that you can use to run the app yourself.

The written tutorial does a much better job of introducing `shiny` than an examples file would do. For each problem or sub-problem, I will provide you with the lesson(s) that you will be required to read in order to be able to finish the problem or sub-problem.

We have a zipped directory with "starter code" for this project here or on scholar:

`/class/datamine/data/spring2020/stat29000project09template.zip`.

Unzip the contents into your workspace, and solve the questions in the project by modifying the `app.R` and `utils.R` files.

After each problem, we've provided you with a list of keywords. These keywords could be package names, functions, or important terms that will point you in the right direction when trying to solve the problem, and give you accurate terminology that you can further look into online. You are not required to utilize all the given keywords. You will receive full points as long as your code gives the correct result.

Question 1: I'd rather be Shiny

Install the following packages:

```
install.packages("shiny")
install.packages("DT")
install.packages("markdown")
install.packages("openssl")
install.packages("bcrypt")
```

```
install.packages("data.table")
install.packages("DiagrammeR")
```

Before attempting any question, you may run the app, and see what the template looks like as an app. If you read `app.R` and `utils.R`, you will find that we already provide you the skeleton code with comments pointing out what is expected. Different parts of this project will ask you to fill in sections of code in each tab, to complete the app!

Important note: It is highly recommended that you use https://rstudio.scholar.rcac.purdue.edu/. Use another system at your own risk. The version of RStudio on https://desktop.scholar.rcac.purdue.edu/ is 99.9.9, and is known to have some strange issues when running code chunks.

The submission for this project is a series of files used for running a `shiny` app that teaches the basics of authentication with usernames and passwords. We are providing you with a "skeleton" of the app in `stat29000project09template.zip`. Copy, paste, and unzip the contents into your own workspace. Modify the `app.R` and `utils.R` files based on the questions below. The following files should be uploaded and submitted as your solution: `overview.md` (unmodified), `step(1-5).md` (all 5 unmodified), `utils.R` (modified), and `app.R` (modified).

Item(s) to submit:

- A directory containing the following files: `overview.md` (unmodified), `step(1-5).md` (all 5 unmodified), `utils.R` (modified), and `app.R` (modified).

1a. *(1 pt)* Include the provided markdown files in the `sidebarPanel` for each `tabPanel`. There is a single markdown file for each of the 6 tabs: `Overview`, `1. Signing up`, `2. Hashing`, `3. Salting`, `4. Storing`, `5. Logging in`.

Note that these markdown files explain what is happening in each of the steps. It would be beneficial to read them.

1b. *(1 pt)* In step (1) you can see that we have a `textInput` for both a username and password, called `username1` and `password1` respectively. Create 4 outputs: `output$p1`, `output$p2`, `output$u1`, `output$u2`. In `shiny`, each output can only be displayed a single time. In our app, we want both `password1` and `username1` to be able to be displayed in two locations. Use `renderText` to create the 4 outputs where `output$p1` and `output$p2` both display the `password1`, and `output$u1` and `output$u2` both display the `username1`.

1c. *(1 pt)* Run the app. In step (2) you should now see your username and password entered in step (1) displayed in step (2)'s `tabPanel`. Create `output$passhash` (that is, an output called `passhash`) using `renderText` that returns the `sha256` hash of the password from step (1). Use the `sha256` function from the `openssl` package. Run the app again and you should now see the hash displayed in step (2) as well!

Hint: Read `step2.md` to see how to use the function `sha256`.

1d. *(1 pt)* Run the app. In step (3) you should see the username and password displayed as well as 4 other labels that are missing output, as well as an `actionButton` that says "New salt". Look at the `eventReactive` in the server function. What this `eventReactive` does is waits for a click of the `newsalt actionButton`. Upon clicking the button, it uses the `gensalt` function from the `bcrypt` library to generate a new salt, and returns this value anywhere the `eventReactive` is called. Create a new output called `salt` using `renderText` that displays the salt when the button is clicked. Run the app. Upon clicking the button in step (3), a salt should appear!

1e. *(1 pt)* Fill in the function `salt_password` in `utils.R` to append the given `salt` argument to the given `password` argument.

Keywords: *paste0*

1f. *(2 pts)* Fill in the function `hash_password` in `utils.R` to create the final hash. The final hash is created by:

1. Salting your password using the function in (1e).
2. Run your salted password through the `sha256` function.
3. Prepend the provided salt from the `salt` argument to the result of (2).

Return this final hash!

Keywords: *paste0*

1g. *(2 pts)* Fill in the function `check_password` in `utils.R` to check if a provided password will match the provided hash once it too has been hashed. To do so:

1. Extract the salt from the beginning of the provided hash (`salted_hash`). The salt corresponds to the first 29 characters of the provided hash.
2. Use the `hash_password` function using the extracted salt from (1) to hash the provided password. Call this result `new_hash`.
3. Return `TRUE` if the `new_hash` matches the hash provided to the `check_password` function. Return `FALSE` otherwise.

Keywords: *substr*

1h. *(1 pt)* If you completed the tasks correctly, steps (4) and (5) should work! Test it out and play with it. In general, is this how you thought passwords work? Write 1-2 sentences commenting on anything you thought was interesting or different, and if you were able to learn anything about `shiny` from the provided R code.

Item(s) to submit:
- 1-2 sentences as comments at the bottom of your `app.R` file.

1i. *(optional)* Also, feel free to improve the appearance of the app.

Project Submission:

Submit your solutions for the project at this URL: https://classroom.github.com/a/p75IMKqQ using the instructions found in the GitHub Classroom instructions folder on Blackboard.

Important note: The submission for this project is a series of files used for running a `shiny` app that teaches the basics of authentication with usernames and passwords. We are providing you with a "skeleton" of the app in `stat29000project09template.zip`. Copy, paste, and unzip the contents into your own workspace. Modify the `app.R` and `utils.R` files based on the questions below. The following files should be uploaded and submitted as your solution: `overview.md` (unmodified), `step(1-5).md` (all 5 unmodified), `utils.R` (modified), and `app.R` (modified).