Midterm (CS 5610)

Benakana Harikrishna Reddy

2/28/2024

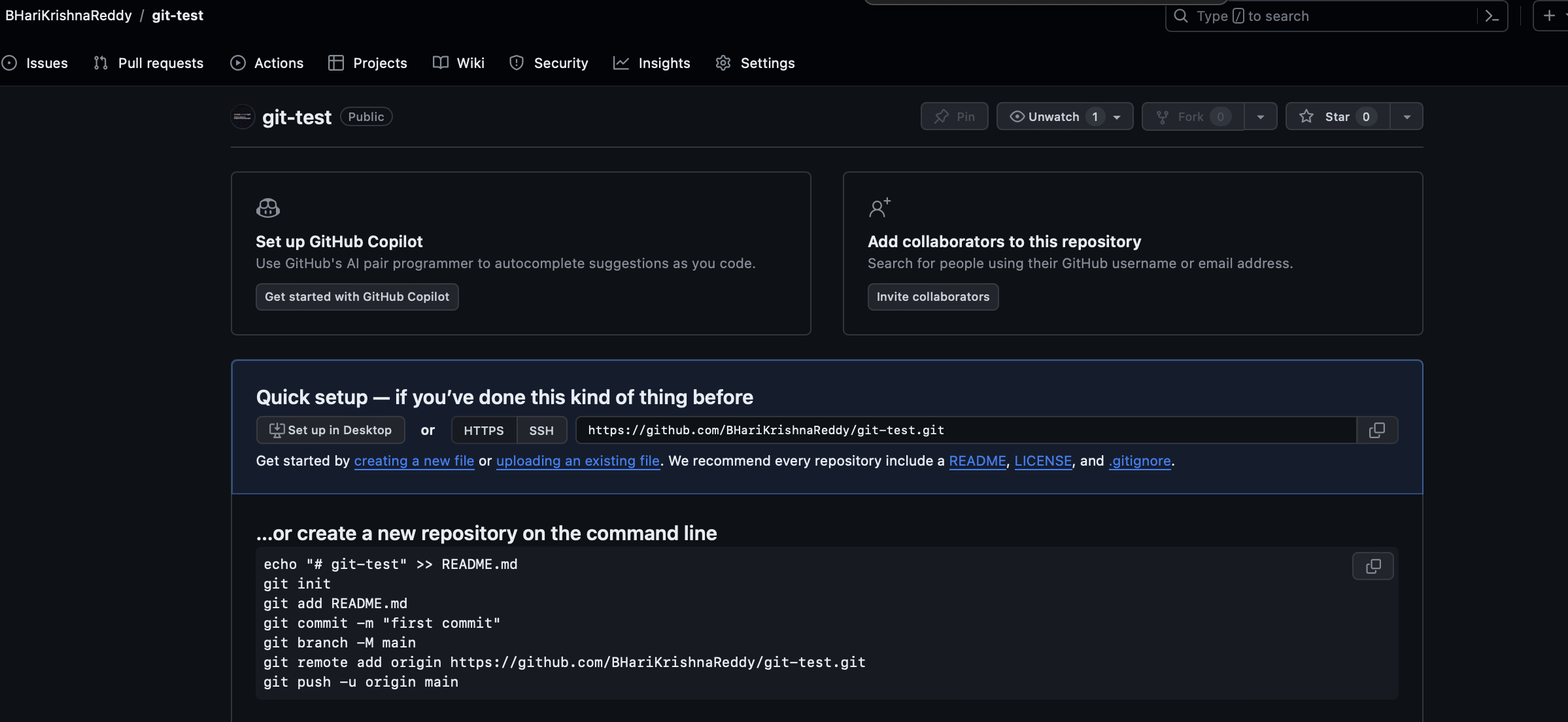
## Insturctions:

1. Download the Midterm.Rmd file from the course eLearning.
2. Open Midterm.Rmd in RStudio.
3. Replace the author name with your name.
4. Write your answer to each problem by editing Midterm.Rmd.
5. After you finish all the problems, click Knit to PDF to create a pdf file or a ‘word\_document’ file. Upload your Midterm.Rmd and pdf/word file to Exam 1 Dropbox in the course eLearning.
6. I certify here that the work on this exam is solely mine. I did not receive any assistance from others, and I did not provide any assistance to others.

PRINT YOUR NAME: Benakana Harikrishna Reddy DATE: 2/28/2024

### Problem 1. For this proplem you will use git to clone and edit a repository (20 pts.)

###### (1) On your Github account, create a new repository (git-test). Take a screenshoot and include it as an answer for this question.(3 pts.)

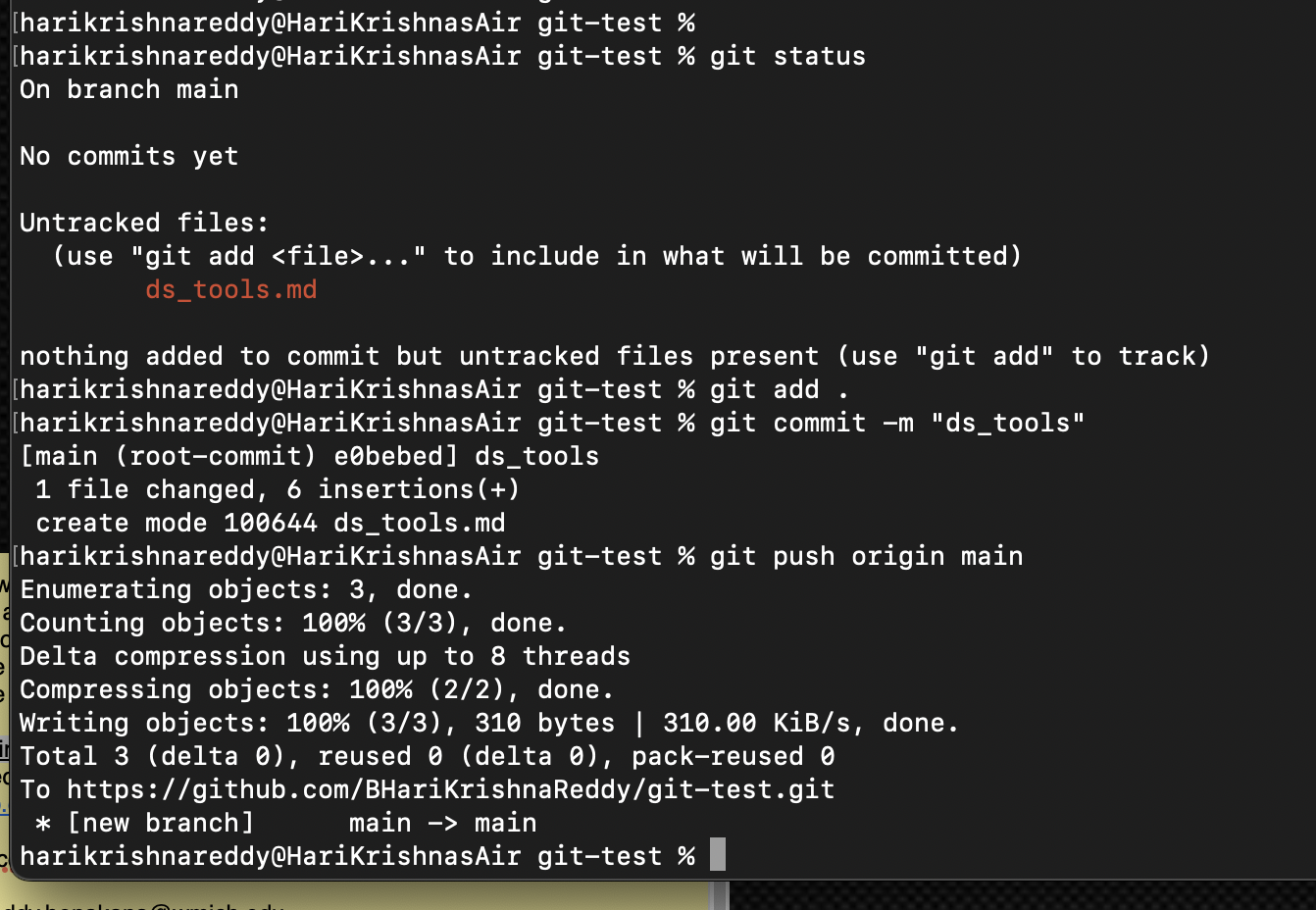


GithubRepo Creation Image

###### (2) Using the terminal, clone **your git-test repository** to your machine (make sure you’re in the desired directory on your terminal).(3 pts.)

# Provide the commands here  
 git clone https://github.com/BHariKrishnaReddy/git-test.git  
 ```  
###### (3) On your machine, create and open up this file (`ds\_tools.md`) in a text editor of your choice.Then, add a list of data science tools. Add this file manually to the `git-test` repository on you machine.(3 pts.)   
```markdown  
1. R Programming  
2. Python  
3. SAS  
4. Jupyter Notebook  
5. MS Excel  
6. Google Analytics

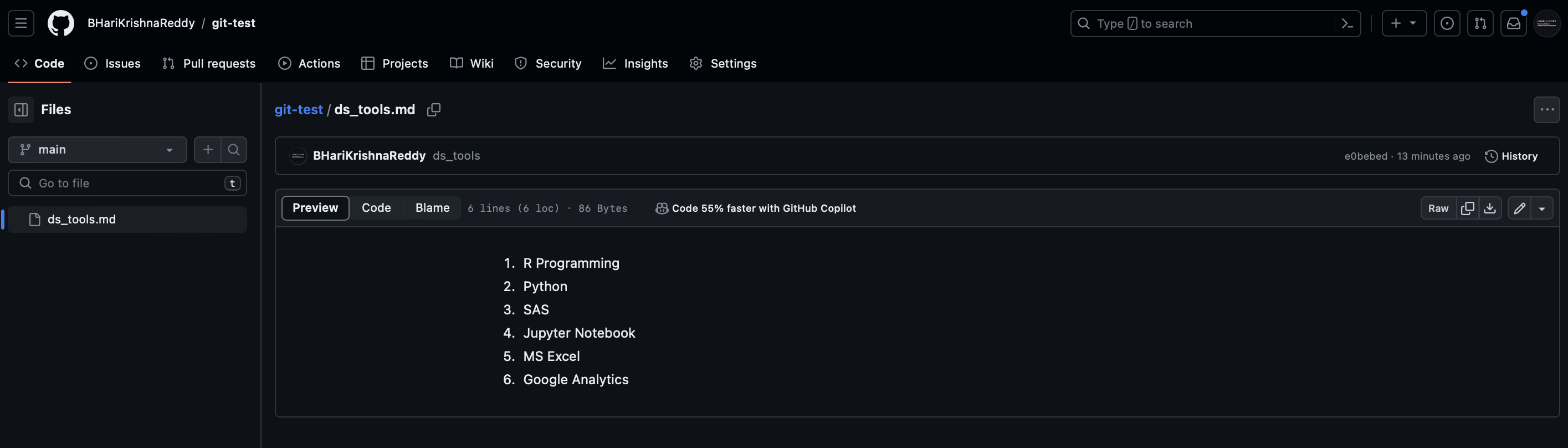
###### (4) Using your terminal, add and commit the changes you’ve made to your repository. Add a screenshot of your terminal.(4 pts.)



GithubRepo Creation Image

# Provide the commands here  
 git status  
 git add .  
 git commit -m "ds\_tools"  
 ```  
###### (5) Push changes up to GitHub, and view them in your web browser. Make sure you are looking at the repository under \_your\_ account! Also, add here a screenshot of your Github repository that contains a new added `ds\_tools.md`.(3 pts.)   
  
![GithubRepo Creation Image](/Users/harikrishnareddy/Desktop/R/WassnaaAl-Mawee/Midterm/local2Remote.png)  
  
```bash  
 # Provide the commands here  
 git push origin main

###### (6) Open up the ds\_tools.md on Github, and add a screenshot of its contents. You should see the list of data science tools. (4 pts.)



GithubRepo commit Image

### Problem 2. Funactional Programming (15 pts.).

###### (1) Consider a list of numeric values. Write a function in R that uses functional programming principles to transform the list by squaring each element, filtering out values greater than 10, and then calculating the sum of the remaining squared values. Discuss the advantages of using functional programming concepts, such as Map, Filter, and Reduce, in solving this problem compared to traditional iterative approaches.

numeric\_values <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)  
transform\_and\_sum <- function(nums) {  
 squared\_values <- lapply(nums, function(x) x^2)  
 filtered\_values <- Filter(function(x) x <= 10, squared\_values)  
 sum\_values <- Reduce(function(x, y) x + y, filtered\_values)  
   
 return(sum\_values)  
}  
  
result <- transform\_and\_sum(numeric\_values)  
print(result)

## [1] 14

# Provide your code/discussion here

### Problem 3. Data Preprocessing and Exploration in R (15 pts.).

##### (1) Imagine you are given a sample dataset containing various variables. Discuss the key steps you would take in R for data preprocessing and exploration. How would you preprocess the data? Additionally, what visualization techniques in R would you employ to gain insights into the data’s distribution, relationships, and potential patterns?

# Provide your code/discussion here  
  
# Loading the Dataset: Begin by loading the housing dataset into R using functions like read.csv() or read.table().  
# Inspecting the Dataset: Use functions like head(), summary(), str(), and dim() to understand the structure, summary statistics, and dimensions of the dataset.  
# Handling Missing Values: Identify missing values and handle them using techniques such as imputation, deletion, or advanced methods like predictive modeling.  
# Handling Outliers: Detect and handle outliers using visualization techniques (e.g., box plots, scatter plots) or statistical methods (e.g., z-score, IQR).  
# Feature Scaling/Normalization: Scale or normalize numerical features to bring them into a similar scale, preventing dominance of certain features in the analysis.  
# Encoding Categorical Variables: Convert categorical variables into numerical representations using techniques like one-hot encoding or label encoding.  
# Feature Engineering: Create new features based on existing ones or domain knowledge to improve model performance.  
# Splitting the Dataset: Split the dataset into training and testing sets to evaluate model performance effectively.  
# Exploratory Data Analysis (EDA):  
# Univariate Analysis: Visualize the distribution of individual variables using histograms, density plots, and summary statistics.  
# Bivariate Analysis: Explore relationships between pairs of variables using scatter plots, correlation matrices, and box plots.  
# Multivariate Analysis: Investigate interactions between multiple variables using techniques like dimensionality reduction (e.g., PCA) or visualization of higher-dimensional spaces.  
  
# Visualization Techniques: Employ various visualization techniques in R to gain insights into the data's distribution, relationships, and potential patterns:  
# Histograms and Density Plots: To visualize the distribution of numerical variables.  
# Box Plots: To compare the distribution of numerical variables across different categories.  
# Scatter Plots: To explore relationships between two numerical variables.  
# Correlation Matrices: To visualize the correlation between numerical variables.  
# Heatmaps: To visualize relationships between variables using color gradients.  
# Pair Plots: To visualize relationships between multiple variables in a pairwise manner.

### Problem 4. Interactive Shiny Apps (30 pts.)

# Use the `diamonds` data set provided by ggplot2  
# Use dplyr's `sample\_n()` function to get a random 1000 rows from the data set  
# Store this sample in a variable `diamonds\_sample`  
  
  
# For convenience store the `range()` of values for the `price` column  
# (of your sample)  
  
  
# For convenience, get a vector of column names from the `diamonds` data set to  
# use as select inputs  
  
  
# To help keep the code organized, we'll store some UI elements in variables  
# \_before\_ defining the UI.  
  
# Define a variable `price\_input` that is a `sliderInput()` with the following  
# properties:  
# - an inputId of `price\_choice`  
# - a label of "Price (in dollars)"  
# - min and max valuesvalue based on the `price\_range` calculated above  
# - a current value equal to the price range  
  
  
# Define a variable `feature\_input` that is a `selectInput()` with the  
# label "Feature of Interest". This dropdown should let the user pick one of  
# the columns of the diamond data set. Use the `carat` column as a default  
# Make sure to set an inputId to reference in your server!  
  
  
# Define a UI using a `fluidPage()` layout with the following content:  
  
 # A `titlePanel` with the title "Diamond Viewer"  
  
 # Your `prince\_input`  
  
 # Your `feature\_input`  
  
 # A `checkboxInput()` labeled "Show Trendline". It's default value is TRUE  
   
  
 # A plotOutput showing the 'plot' output (based on the user specifications)  
   
  
  
# Define a `server` function (with appropriate arguments)  
# This function should perform the following:  
  
 # Assign a reactive `renderPlot()` function to the outputted 'plot' value  
   
 # This function should first filter down the `diamonds\_sample` data  
 # using the input price range (remember to get both ends)!  
   
  
 # Use the filtered data set to create a ggplot2 scatter plot with the  
 # user-select column on the x-axis, and the price on the y-axis,  
 # and encode the "cut" of each diamond using color  
 # Save your plot as a variable.  
   
 # Finally, if the "trendline" checkbox is selected, you should add (+)  
 # a geom\_smooth geometry (with `se=FALSE`) to your plot  
 # Hint: use an if statement to see if you need to add more geoms to the plot  
  
   
 # Be sure and return the completed plot!  
   
  
# Create a new `shinyApp()` using the above ui and server  
  
# Hint: You must submit an app.R for this question. Your app must match the given example.png

### Problem 5. State which of the follwing statement is TRUE and which is FALSE (20 pts.)

1. We say the dataset is tidy when it looks like below.

subject\_id <- c(1, 2, 3)  
gender <- c("Male", "Female", "Male")  
height <- c(170, 160, 175)  
weight <- c(65, 55, 70)  
  
# Creating a data frame  
non\_tidy\_data <- data.frame(subject\_id, gender, height, weight)

True

1. The values 3, ‘apple’, and TRUE belong to the complex data type in R. False
2. We use geom\_bar() function to plot a histogram for continuous variables. False
3. mutate() function creates additional columns for a data frame. True
4. The leaflet() function in the leaflet package is used to create a basic map, allowing you to specify the initial center coordinates and zoom level, among other options. True
5. Anonymous functions in R, created using the function keyword without assigning a name, are also known as lambda functions and are often used for short-term, one-off operations. True
6. Comments in SQL are written on their own line surrounded by /\* \*/ False
7. In R, the httr package is commonly used for making HTTP requests to APIs, and the GET function is primarily used for sending data to the server. False
8. A foreign key is used to establish a link between two tables in a relational database. True
9. In a Shiny app, the server.R file is responsible for defining the user interface (UI), while the ui.R file handles server-side logic and data processing. False