1. Installation of packages

Code:

#install.packages("rvest")

#install.packages("dplyr")

* These lines are commented out. If you are running this code for the first time, you need to install the required packages. You can uncomment and run these lines to install the rvest and dplyr packages if they aren't already installed.

1. Importing Libraries

Code:

library(rvest)

library(dplyr)

* **library(rvest):** This line loads the rvest package which is used for web scraping in R. It allows you to easily scrape web pages by extracting their HTML content.
* **library(dplyr):** This line loads the dplyr package which is used for data manipulation. It's often used to transform and summarize data in R.

1. Initialize dataframe

Code:

hotel\_review\_dataset = data.frame()

* Initializes an empty data frame called hotel\_review\_dataset. This data frame will later hold the scraped data (reviewer names and their reviews).

1. Define page link

Code:

link = "your page link here"

* This line defines a variable link which should contain the URL of the webpage you want to scrape. Replace "your page link here" with the actual URL of the hotel review page.

1. Read the html content

Code:

hotel = read\_html(link)

* This line uses the read\_html() function from the rvest package to read the HTML content of the webpage located at the link. The HTML structure of the webpage is stored in the hotel variable.

1. Extract “Reviewer Name” data

Code:

reviewer\_name = hotel %>% html\_nodes(".uyyBf") %>% html\_text()

* This line extracts the reviewer names from the HTML content.
* **hotel:** The webpage content (stored as HTML).
* **%>%:** The pipe operator that passes the previous value (in this case, hotel) to the next function.
* **html\_nodes(".uyyBf"):** This function finds all the HTML nodes (or elements) that match the given CSS selector .uyyBf. Replace .uyyBf with the correct CSS class or identifier for the reviewer names.
* **html\_text():** This extracts the text content from the selected HTML nodes (i.e., the names of the reviewers).

1. Extract “Review” data

Code:

review = hotel %>% html\_nodes(".vTVDc > .FKffI .\_T") %>% html\_text()

* This line extracts the review text using similar logic.
* **html\_nodes(".vTVDc > .FKffI .\_T"):** This selects the HTML nodes containing the reviews, using the specified CSS selector (**Selector Gadget**). Replace .vTVDc > .FKffI .\_T with the correct CSS selector for the reviews.
* **html\_text():** This extracts the text content (the actual review) from the selected nodes.

1. Create dataset

Code:

hotel\_review\_dataset = data.frame(reviewer\_name, review, stringsAsFactors = FALSE)

* Creates a data frame hotel\_review\_dataset containing two columns:
* **reviewer\_name:** A column of reviewer names (from the previous step).
* **review:** A column of corresponding reviews.
* **stringsAsFactors = FALSE:** This ensures that the text data (characters) remains as strings, and not as factors, which is a different data type in R.

1. Convert data frame to csv file

Code:

write.csv(hotel\_review\_dataset, "The CSV File Name.csv")

* This line writes the hotel\_review\_dataset data frame to a CSV file.
* The CSV file will be saved with the name "The CSV File Name.csv". Replace this with an appropriate filename.
* This is the final output of the code, storing all scraped reviews and reviewer names into a CSV file.

Summary:

1. It loads necessary libraries for web scraping and data manipulation.
2. Defines the URL of the hotel review webpage.
3. Extracts reviewer names and review texts using appropriate CSS selectors.
4. Stores the scraped data into a data frame.
5. Saves the data frame as a CSV file.

**Note:** Make sure to replace the CSS selectors (like .uyyBf, .vTVDc > .FKffI .\_T) with the correct ones for your specific webpage, which you can get using tools like the Selector Gadget (a browser extension that helps you find CSS selectors).