Experiment based on R-JSON andWeb

AIM:

Experiment based on R-JSON andWeb

THEORY:

JSON (JavaScript Object Notation) is a lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate. It is widely used for transmitting data in web applications between a server and a client.

R is a programming language and environment commonly used for statistical computing and graphics. It has several packages that facilitate working with JSON data and interacting with web APIs.

Web APIs (Application Programming Interfaces) allow different software applications to communicate with each other over the internet. They often return data in JSON format, making it easy to integrate with R

PRACTICAL 6 :

CODE :-

Create employee.json file with the following content:

```
[
  {
    "Name": "John",
    "Age": 30,
    "Department": "Finance",
    "Salary": 80000
  },
  {
    "Name": "Sarah",
    "Age": 28,
    "Department": "HR",
    "Salary": 75000
```

```
  },
  {
    "Name": "Mike",
    "Age": 35,
    "Department": "IT",
    "Salary": 85000
  }
]
```

Code:

```
# Load required libraries
library(jsonlite)
library(httr)
library(dplyr)  # For data manipulation

# Define the path to the external JSON file
json_file_path <- "employees.json"  # Ensure the file exists in your working directory

# Read JSON data from the file
json_data <- fromJSON(json_file_path)

# Print original JSON data
print("Original JSON Data:")
print(json_data)

# Extract specific fields
print("Extracted Names:")
print(json_data$Name)

print("Extracted Departments:")
```

```
print(json_data$Department)

# Modify Data: Increase Salary by 10%
json_data$Salary <- json_data$Salary * 1.10

# Add a new field (e.g., "Experience" with random years)
set.seed(123)
json_data$Experience <- sample(3:10, nrow(json_data), replace = TRUE)

# Convert JSON to a DataFrame
json_df <- as.data.frame(json_data)
print("Converted JSON to DataFrame:")
print(json_df)

# Filter employees earning more than 80,000
high_salary_employees <- json_df %>% filter(Salary > 80000)
print("Employees earning more than 80,000:")
print(high_salary_employees)

# Write modified JSON to a new file
write_json(json_data, "modified_employees.json", pretty = TRUE)

print("Modified JSON data saved to modified_employees.json")

# Fetch JSON data from a web API
url <- "https://jsonplaceholder.typicode.com/users"  # Example API endpoint
response <- GET(url)

# Convert response to text
```

response_text <- content(response, "text", encoding = "UTF-8")

Parse JSON response

response_json <- fromJSON(response_text)

Print API response data

print("API Response:")

print(response_json)

Extract specific details from API data (Example: Extract user names)

user_names <- response_json$name

print("User Names from API:")

print(user_names)

```
# Load required libraries
install.packages("jsonlite")
install.packages("httr")
install.packages("dplyr")
library(jsonlite)
library(httr)
library(dplyr) # For data manipulation
```

```
> # Define the path to the external JSON file
> json_file_path <- "employees.json" # Ensure the file exists in your working directory
>
> # Read JSON data from the file
> json_data <- fromJSON(json_file_path)
>
> # Print original JSON data
> print("Original JSON Data:")
[1] "Original JSON Data:"
> print(json_data)
   Name Age Department Salary
1  John  30    Finance  80000
2 Sarah  28         HR  75000
3  Mike  35         IT  85000
>
> # Extract specific fields
> print("Extracted Names:")
[1] "Extracted Names:"
> print(json_data$Name)
[1] "John"  "Sarah" "Mike"
>
> print("Extracted Departments:")
[1] "Extracted Departments:"
> print(json_data$Department)
[1] "Finance" "HR"      "IT"
>
> # Modify Data: Increase Salary by 10%
> json_data$Salary <- json_data$Salary * 1.10
>
> # Add a new field (e.g., "Experience" with random years)
> set.seed(123)
> json_data$Experience <- sample(3:10, nrow(json_data), replace = TRUE)
>
> # Convert JSON to a DataFrame
> json_df <- as.data.frame(json_data)
> print("Converted JSON to DataFrame:")
[1] "Converted JSON to DataFrame:"
> print(json_df)
   Name Age Department Salary Experience
1  John  30    Finance  88000          9
2 Sarah  28         HR  82500          9
3  Mike  35         IT  93500          5
```

```
> # Filter employees earning more than 80,000
> high_salary_employees <- json_df %>% filter(Salary > 80000)
> print("Employees earning more than 80,000:")
[1] "Employees earning more than 80,000:"
> print(high_salary_employees)
   Name Age Department Salary Experience
1  John  30    Finance  88000          9
2 Sarah  28         HR  82500          9
3  Mike  35         IT  93500          5
>
> # Write modified JSON to a new file
> write_json(json_data, "modified_employees.json", pretty = TRUE)
> print("Modified JSON data saved to modified_employees.json")
[1] "Modified JSON data saved to modified_employees.json"
>
> # Fetch JSON data from a web API
> url <- "https://jsonplaceholder.typicode.com/users" # Example API endpoint
> response <- GET(url)
>
> # Convert response to text
> response_text <- content(response, "text", encoding = "UTF-8")
>
> # Parse JSON response
> response_json <- fromJSON(response_text)
```

```
> # Print API response data
> print("API Response:")
[1] "API Response:"
> print(response_json)
   id                     name         username                     email    address.street
1   1            Leanne Graham             Bret         Sincere@april.biz       Kulas Light
2   2             Ervin Howell         Antonette         Shanna@melissa.tv     Victor Plains
3   3          Clementine Bauch         Samantha        Nathan@yesenia.net Douglas Extension
4   4          Patricia Lebsack         Karianne Julianne.OConner@kory.org       Hoeger Mall
5   5          Chelsey Dietrich            Kamren  Lucio_Hettinger@annie.ca      Skiles Walks
6   6      Mrs. Dennis Schulist Leopoldo_Corkery   Karley_Dach@jasper.info Norberto Crossing
7   7           Kurtis Weissnat     Elwyn.Skiles    Telly.Hoeger@billy.biz         Rex Trail
8   8 Nicholas Runolfsdottir V    Maxime_Nienow      Sherwood@rosamond.me  Ellsworth Summit
9   9           Glenna Reichert          Delphine   Chaim_McDermott@dana.io        Dayna Park
10 10       Clementina DuBuque   Moriah.Stanton    Rey.Padberg@karina.biz   Kattie Turnpike
   address.suite   address.city address.zipcode address.geo.lat address.geo.lng
1       Apt. 556    Gwenborough      92998-3874        -37.3159         81.1496
2      Suite 879    Wisokyburgh      90566-7771        -43.9509        -34.4618
3      Suite 847  McKenziehaven      59590-4157        -68.6102        -47.0653
4       Apt. 692    South Elvis      53919-4257         29.4572       -164.2990
5      Suite 351     Roscoeview           33263        -31.8129         62.5342
6       Apt. 950  South Christy      23505-1337        -71.4197         71.7478
7      Suite 280      Howemouth      58804-1099         24.8918         21.8984
8      Suite 729      Aliyaview           45169        -14.3990       -120.7677
9      Suite 449 Bartholomebury      76495-3109         24.6463       -168.8889
10     Suite 198    Lebsackbury      31428-2261        -38.2386         57.2232
                  phone       website       company.name                      company.catchPhrase
1  1-770-736-8031 x56442 hildegard.org    Romaguera-Crona   Multi-layered client-server neural-net
2    010-692-6593 x09125 anastasia.net       Deckow-Crist           Proactive didactic contingency
3          1-463-123-4447   ramiro.info Romaguera-Jacobson        Face to face bifurcated interface
4       493-170-9623 x156      kale.biz      Robel-Corkery Multi-tiered zero tolerance productivity
5           (254)954-1289  demarco.info        Keebler LLC     User-centric fault-tolerant solution
6   1-477-935-8478 x6430       ola.org  Considine-Lockman       Synchronised bottom-line interface
7            210.067.6132      elvis.io        Johns Group       Configurable multimedia task-force
8      586.493.6943 x140  jacynthe.com    Abernathy Group            Implemented secondary concept
9    (775)976-6794 x41206    conrad.com      Yost and Sons    Switchable contextually-based project
10           024-648-3804   ambrose.net         Hoeger LLC        Centralized empowering task-force
```

```
                          company.bs
1          harness real-time e-markets
2      synergize scalable supply-chains
3       e-enable strategic applications
4  transition cutting-edge web services
5      revolutionize end-to-end systems
6      e-enable innovative applications
7         generate enterprise e-tailers
8         e-enable extensible e-tailers
9      aggregate real-time technologies
10            target end-to-end models
>
> # Extract specific details from API data (Example: Extract user names)
> user_names <- response_json$name
> print("User Names from API:")
[1] "User Names from API:"
> print(user_names)
 [1] "Leanne Graham"           "Ervin Howell"            "Clementine Bauch"
 [4] "Patricia Lebsack"        "Chelsey Dietrich"        "Mrs. Dennis Schulist"
 [7] "Kurtis Weissnat"         "Nicholas Runolfsdottir V" "Glenna Reichert"
[10] "Clementina DuBuque"
```

Conclusion :

- **JSON and APIs**: JSON is a widely used format for data exchange in web applications, and R provides powerful tools for working with JSON data and web APIs.
- **Data Manipulation**: The combination of **dplyr** for data manipulation and **ggplot2** for visualization makes R a robust choice for data analysis tasks.
- **Real-World Applications**: This approach can be applied to various domains, such as finance, social media, and health, where data is often accessed via APIs

For Faculty Use

Correction Parameters	Formative Assessment [40%]	Timely completion of Practical [40%]	Attendance / Learning Attitude [20%]	
Marks Obtained				