Program for Web Scraping and Data Extraction:

- Install and Load Required Packages:
    - Install the necessary packages for web scraping, such as rvest, xml2, and httr.
    - Load the packages into your R environment using the library() function.
- Specify the Target Website and URL:
    - Identify the website you want to scrape data from.
    - Define the URL of the specific webpage or API endpoint containing the data you need.
- Send HTTP Requests and Handle Authentication (if required):
    - Use the GET() or POST() functions from the httr package to send HTTP requests to the website.
    - Set headers, parameters, or authentication credentials as needed.
- Retrieve HTML Content and Parse XML/HTML:
    - Use the GET() function to retrieve the HTML content of the webpage.
    - Parse the HTML content using the read_html() function from the rvest package.
- Extract Data from HTML:
    - Inspect the HTML structure of the webpage to identify the elements and attributes containing the desired data.
    - Use functions such as html_nodes() and html_text() from the rvest package to extract specific elements or text from the HTML content.
    - Apply CSS selectors or XPath expressions to target specific elements.
- Perform Data Cleaning and Transformation:
    - Clean the extracted data by removing unwanted characters, handling missing values, or applying regular expressions.
    - Convert the extracted data into appropriate data structures (e.g., data frames, lists, or vectors) for further analysis or storage.
- Save or Analyse the Extracted Data:
    - Save the extracted data to a file (e.g., CSV or Excel) using R functions like write.csv() or write.xlsx().
    - Perform further data analysis or visualization on the extracted data using appropriate R packages and techniques.

```
# Install and Load Required Packages

#install.packages(c("rvest", "xml2", "httr"))


library(rvest)

library(xml2)

library(httr)

# Define the URL

#url <- "https://www.example.com"


# If authentication is required, set your username and password

#username <- "your_username"
```

```
#password <- "your_password"

# Send a GET request
#response <- GET(url, authenticate(username, password))
# Specify the Target Website and URL
url <-
"https://en.wikipedia.org/wiki/List_of_countries_and_dependencies_and_their_capitals_in_native_languages"

# Send HTTP Requests and Handle Authentication (if required)
response <- tryCatch(GET(url), error = function(e) e)

# Check if connection was successful
if (!inherits(response, "error")) {
  cat("Successfully connected to the website.\n")

  # Retrieve HTML Content and Parse XML/HTML
  html_content <- read_html(response$content, encoding = "UTF-8")

  # Extract Data from HTML
  # Extracting country names
  countries <- html_content %>%
    html_nodes(".wikitable tbody tr td:nth-child(2)") %>%
    html_text()

  # Extracting capital names
  capitals <- html_content %>%
    html_nodes(".wikitable tbody tr td:nth-child(3)") %>%
    html_text()

  # Perform Data Cleaning and Transformation
  # Remove unnecessary elements like references and decode HTML entities
  clean_countries <- gsub("\\[.*?\\]", "", countries)
  clean_countries <- gsub("&nbsp;", " ", clean_countries)

  clean_capitals <- gsub("\\[.*?\\]", "", capitals)
  clean_capitals <- gsub("&nbsp;", " ", clean_capitals)

  # Remove Unicode characters and symbols
  clean_countries <- enc2utf8(iconv(clean_countries, "UTF-8", "ASCII", sub = " "))
  clean_capitals <- enc2utf8(iconv(clean_capitals, "UTF-8", "ASCII", sub = " "))

  # Combine data into a data frame
  country_capitals <- data.frame(Country = clean_countries, Capital = clean_capitals)

  # Save or Analyze the Extracted Data
  # Save the extracted data to a CSV file
  write.csv(country_capitals, file = "country_capitals.csv", row.names = FALSE)
} else {
  cat("Failed to connect to the website. Please check your internet connection or the URL.\n")
}
```