



Workflow Essentials

Reproducible Report With Quarto



Dr Muhammad Saufi Abdullah

DrPH (Epidemiology) Candidate

Creating Reports with R Quarto

1.1 Overview of Quarto

Quarto allows you to create documents that combine data analysis, results, and explanations all in one place. It ensures that your work is easily reproducible and can be converted into various formats like **HTML**, **PDF**, and **Word** files.

Quarto can be used in three main ways:

1
“
To share clear conclusions with decision-makers without including all the technical details.
”

2
“
To collaborate with other data researchers by sharing both findings and the underlying code.
”

3
“
As a digital notebook to document the entire data analysis process.
”

Quarto Document

To begin learning **Quarto**, let's create a new working directory and name this project '**Quarto Report**'.

In the **File** menu, click and select **Quarto Document**. Name the document '**Data Analysis**' and choose **HTML** as the output format.

New Quarto Document

Document
Presentation
Interactive

Title: Data Analysis

Author: (optional)

☒ HTML
Recommended format for authoring (you can switch to PDF or Word output anytime)

☐ PDF
PDF output requires a LaTeX installation (e.g. <https://yihui.org/tinytex/>)

☐ Word
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux)

Engine: Knitr

Editor: ☒ Use visual markdown editor ?

? [Learn more about Quarto](#)

Create Empty Document Create Cancel

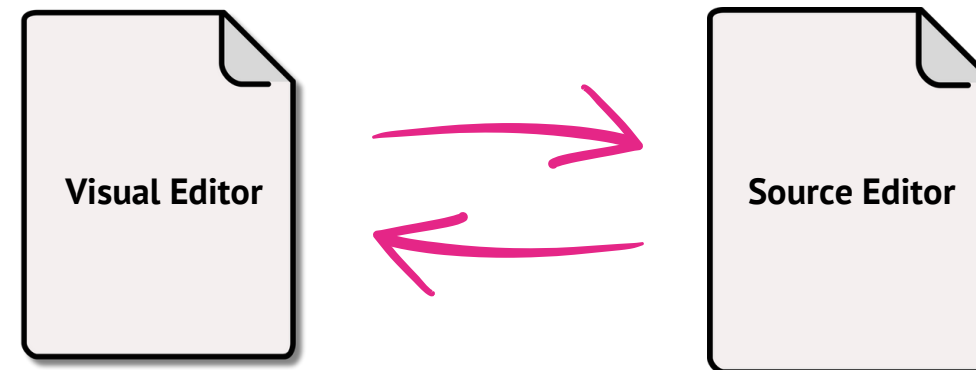
Save the Document

Save the file within the working directory by clicking **`File → Save`** or using the keyboard shortcut **`Ctrl + S`** (Windows) or **`Cmd + S`** (Mac).

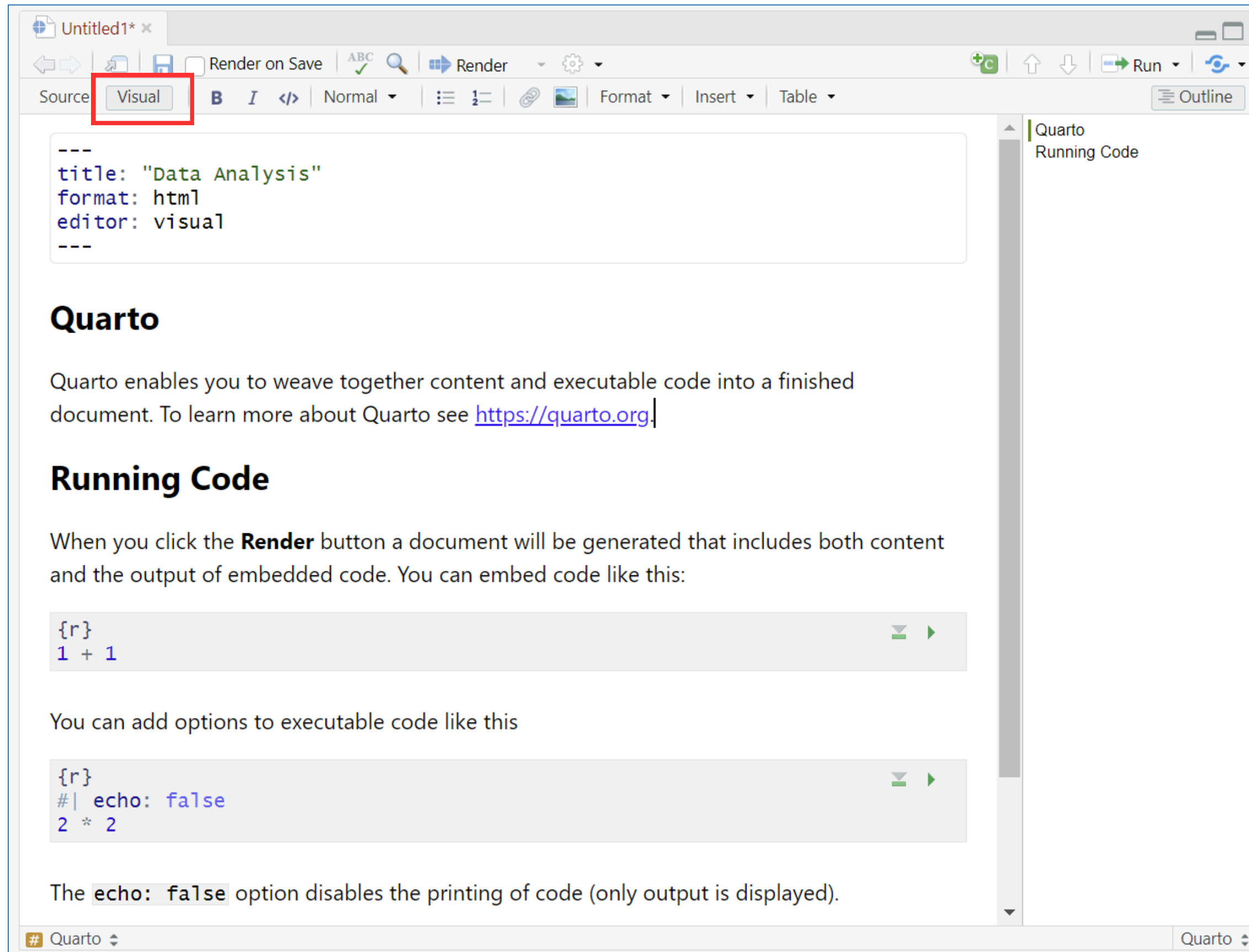
Name the file **`Analysis`**. Note that it will have a **`.qmd`** extension.

Visual and Source Editors

RStudio provides two editors for **Quarto** documents: the **Visual Editor** and the **Source Editor**.

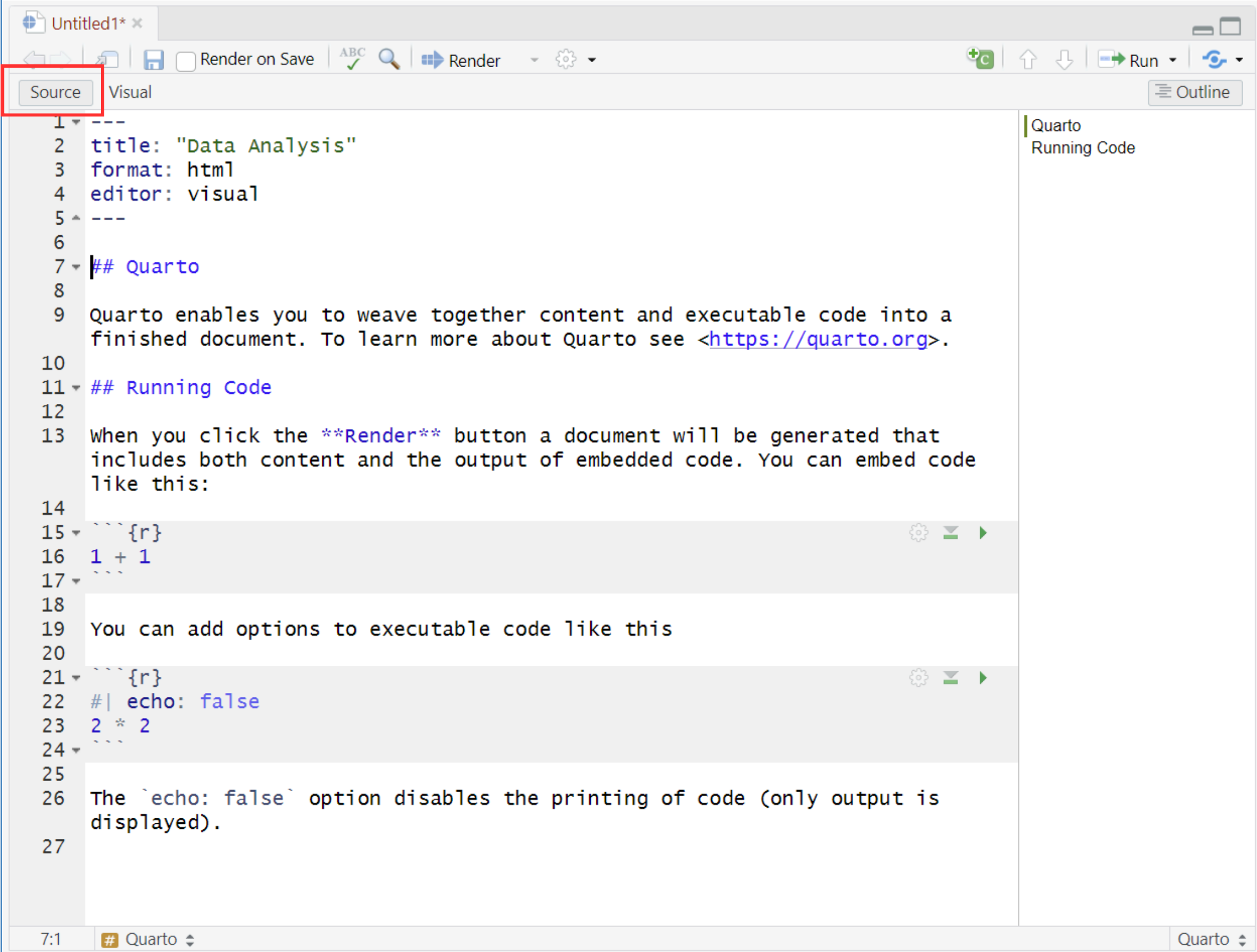


*You can seamlessly switch between these two editors using the buttons at the top left of the **Source** pane.*



***Visual Editor** offers an intuitive interface that allows you to format text and insert elements easily.*

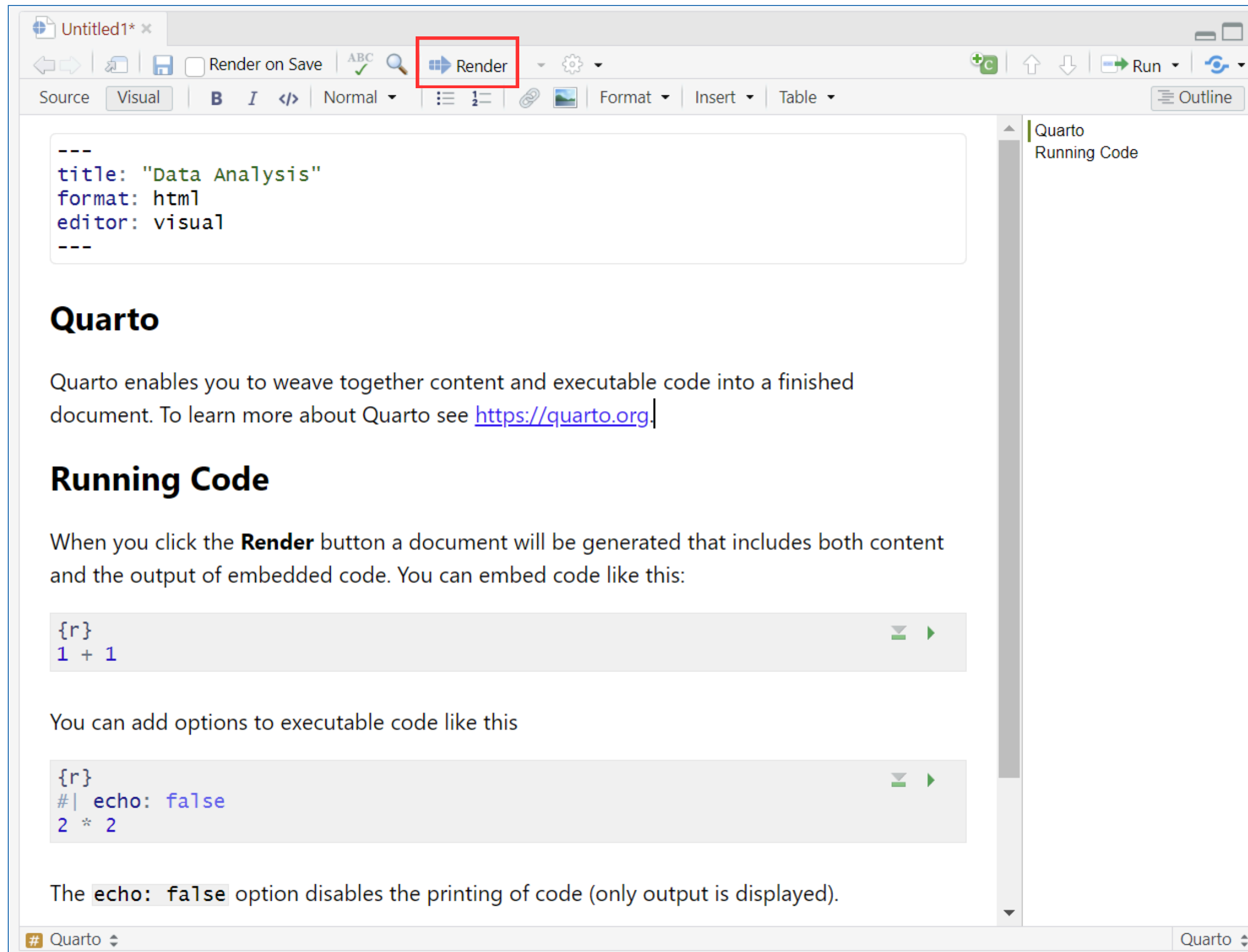
Source Editor, on the other hand, lets you manually enter syntax, giving you finer control over the document's formatting.



The screenshot shows the Quarto Source Editor interface. The top toolbar includes buttons for file operations, a 'Render on Save' checkbox, a 'Render' button, and a 'Run' button. The 'Source' tab is selected, and the document content is as follows:

```
1 ---
2 title: "Data Analysis"
3 format: html
4 editor: visual
5 ---
6
7 ## Quarto
8
9 Quarto enables you to weave together content and executable code into a
10 finished document. To learn more about Quarto see <https://quarto.org>.
11
12 ## Running Code
13
14 When you click the Render button a document will be generated that
15 includes both content and the output of embedded code. You can embed code
16 like this:
17
18 ```{r}
19 1 + 1
20 ```
21
22 You can add options to executable code like this
23
24 ```{r}
25 #| echo: false
26 2 * 2
27 ```
28
29 The `echo: false` option disables the printing of code (only output is
30 displayed).
```

The right sidebar shows 'Quarto Running Code' and an 'Outline' button. The status bar at the bottom indicates '7:1' and the active file is '# Quarto'.



Render the Document

The ``rmarkdown`` package is required to render a **Quarto** file. If the package is not already installed on your machine, you can easily do so. Simply copy the code below, paste it into the **Console** pane, and press **`Enter`**.

In Console pane:

```
install.packages("rmarkdown")
```

By clicking the **`Render`** button at the top of the **Source** pane, you can transform the **Quarto** document into an **HTML** document.

Your browser should automatically open the **HTML** document in a new tab.

If it doesn't, you can find the **HTML** file in the same folder where your **Quarto** document is located.

*You should see the **HTML** document in your browser.*

Data Analysis

Quarto

Quarto enables you to weave together content and executable code into a finished document. To learn more about Quarto see <https://quarto.org>.

Running Code

When you click the **Render** button a document will be generated that includes both content and the output of embedded code. You can embed code like this:

```
1 + 1
```

```
[1] 2
```

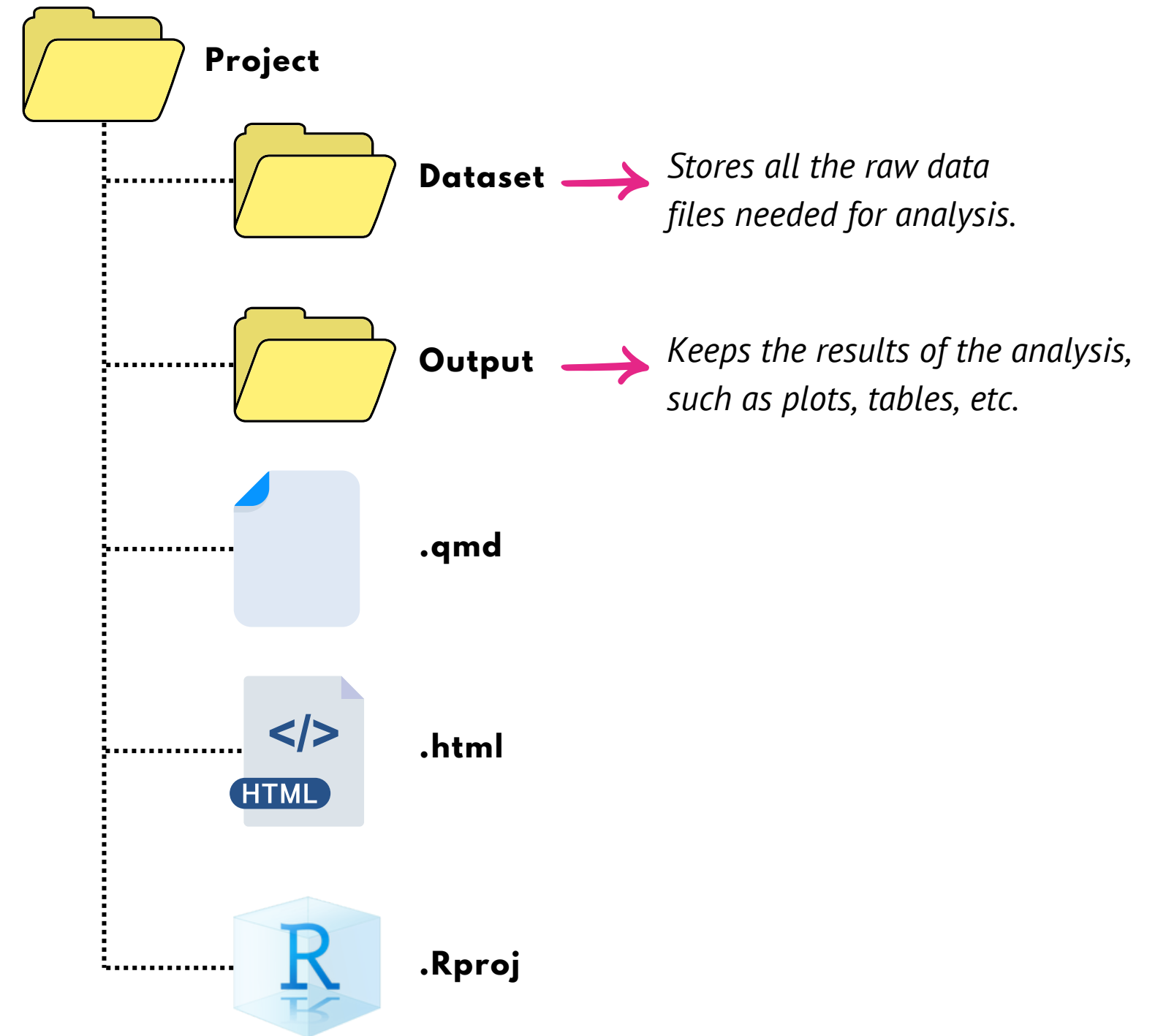
You can add options to executable code like this

```
[1] 4
```

The `echo: false` option disables the printing of code (only output is displayed).

You will find these files within your working directory.

Additionally, you can create **`Dataset`** and **`Output`** folders to organize your datasets and the results of your analysis as needed.



YAML Header

At the beginning of the document, you will notice a block of text enclosed by triple dashes (`---`).

This is the **YAML** header, which defines the settings and configurations for the document.

You can customize this header by adding details such as the author's name and the date.

```
---  
title: "Data Analysis"  
author: "Dr. Muhammad Saufi"  
date: 14 August 2024  
format: html  
editor: visual  
---
```

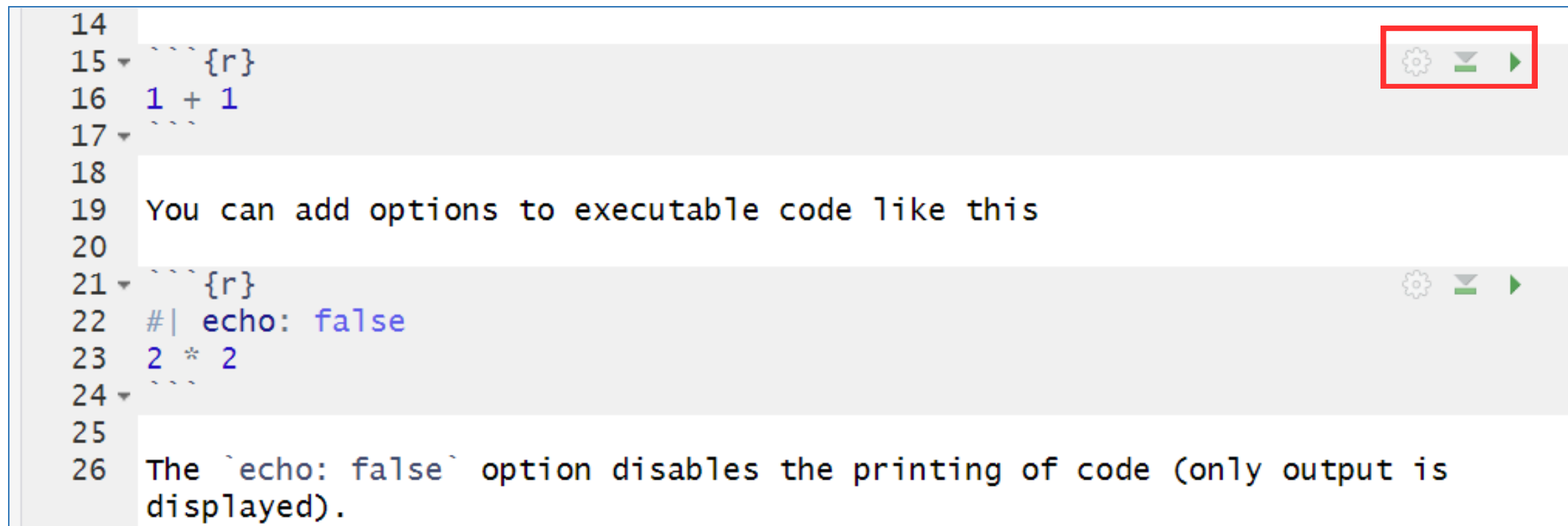
Code Chunks

Code chunks are sections of embedded code that allow you to execute code within **Quarto** document and include the results directly in the output.

To insert a code chunk, click the **Insert** button at the top of the **Source** pane, or use the keyboard shortcut **Ctrl + Alt + I** (Windows) or **Cmd + Option + I** (Mac).

You can run the code in two ways:

- Place the cursor on any line of the code and press **Ctrl + Enter** (Windows) or **Cmd + Enter** (Mac).
- Click the **Arrow** icon at the top right of the code chunk to run the entire chunk.



```
14
15 {r}
16 1 + 1
17
18
19 You can add options to executable code like this
20
21 {r}
22 #| echo: false
23 2 * 2
24
25
26 The `echo: false` option disables the printing of code (only output is displayed).
```

1.2 Data Analysis in Quarto

Let's proceed with performing some data analysis in **Quarto**.

First, delete everything in the **Quarto** document except for the **YAML** header.

Next, download the ``penguins.csv`` dataset from this [link](#) to use in our analysis.

It's important to keep your workflow organized. Here is an example of how to structure your analysis:

Loading the Packages

```
```{r}
library(tidyverse)
```
```

Reading the Dataset

```
```{r}
penguins <- read_csv("Datasets/penguins.csv")
```
```

Exploring the Dataset

```
```{r}
Get a quick overview of dataset
glimpse(penguins)
```
```

1.3 Headings

To organize **Quarto** document effectively, you can create headings by using the `#` symbol.

The number of `#` symbols will determine the heading level; more symbols indicate lower-level headings.

| | | |
|-----------------------------|---|-----------------|
| <code># Header 1</code> | → | Header 1 |
| <code>## Header 2</code> | → | Header 2 |
| <code>### Header 3</code> | → | Header 3 |
| <code>#### Header 4</code> | → | Header 4 |
| <code>##### Header 5</code> | → | Header 5 |
| <code>##### Header 6</code> | → | Header 6 |

*From now on, we will be using the **Source Editor** to better illustrate how coding works in **Quarto**.*

Let's create a structured **Quarto** document with headings and subheadings:

```
# Setting the Environment

## Loading the Packages

```{r}
Load the packages into R session
library(tidyverse)
```

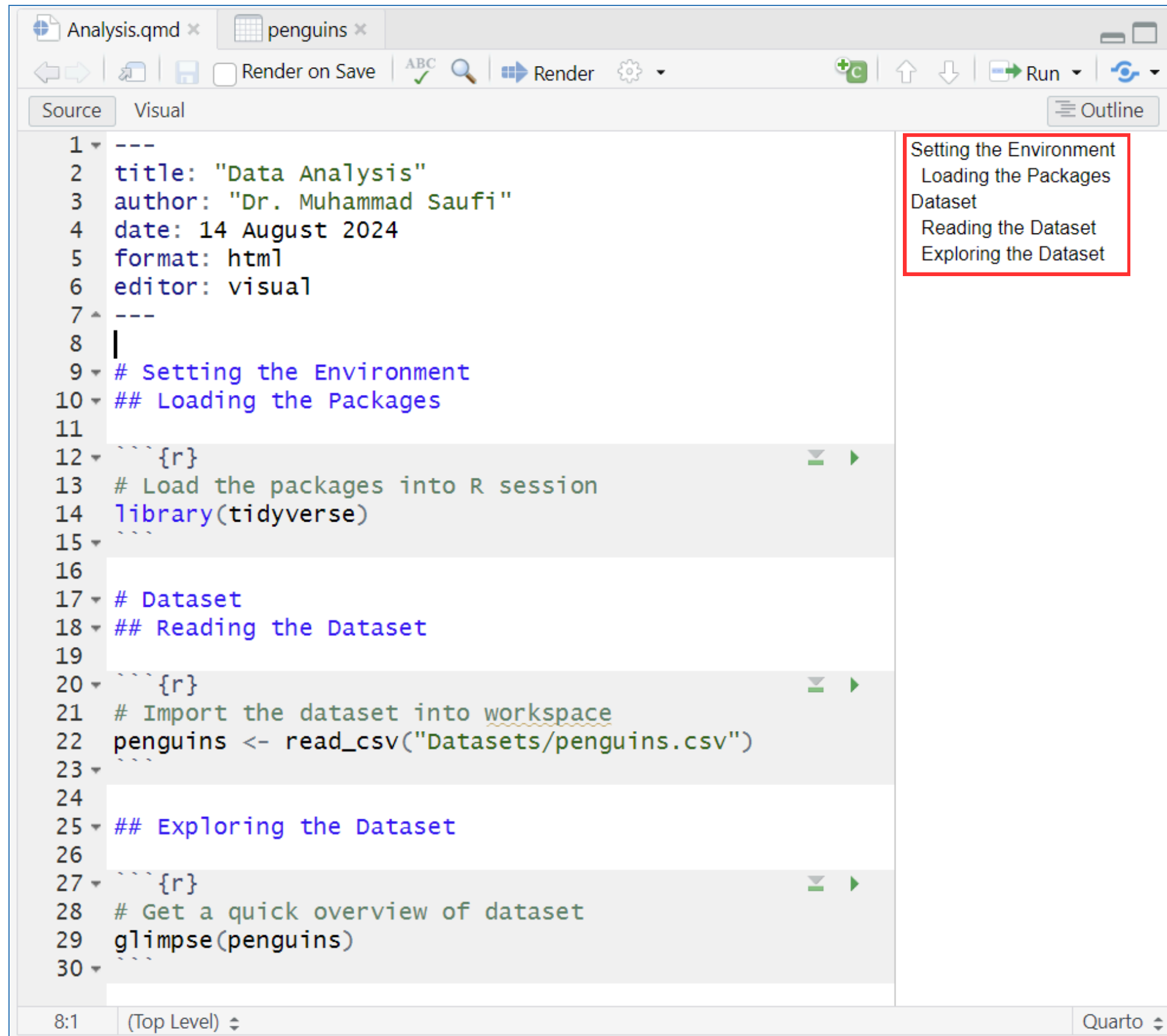
# Dataset

## Reading the Dataset

```{r}
Import the dataset into workspace
penguins <- read_csv("Datasets/penguins.csv")
```

## Exploring the Dataset

```{r}
Get a quick overview of dataset
glimpse(penguins)
```
```



The screenshot shows the Quarto editor interface. The top toolbar includes buttons for navigation, saving, rendering, and running. The main editor area is split into two panes: 'Source' on the left and 'Outline' on the right. The 'Source' pane displays R code with comments indicating the structure of the document. The 'Outline' pane shows a list of sections that can be clicked to navigate to specific parts of the document.

```
1 ---  
2 title: "Data Analysis"  
3 author: "Dr. Muhammad Saufi"  
4 date: 14 August 2024  
5 format: html  
6 editor: visual  
7 ---  
8 |  
9 # Setting the Environment  
10 ## Loading the Packages  
11  
12 ```{r}  
13 # Load the packages into R session  
14 library(tidyverse)  
15 ```  
16  
17 # Dataset  
18 ## Reading the Dataset  
19  
20 ```{r}  
21 # Import the dataset into workspace  
22 penguins <- read_csv("Datasets/penguins.csv")  
23 ```  
24  
25 ## Exploring the Dataset  
26  
27 ```{r}  
28 # Get a quick overview of dataset  
29 glimpse(penguins)  
30 ```
```

The 'Outline' panel on the right lists the following sections:

- Setting the Environment
- Loading the Packages
- Dataset
- Reading the Dataset
- Exploring the Dataset

The status bar at the bottom indicates the current position is at line 8:1, (Top Level).

The headings will appear under the **Outline** tab. You can click on these headings to directly navigate to the selected section. This is especially useful if you have a long document.

Note that if you include a `#` symbol followed by text within a code chunk, **Quarto** will recognize it as a comment instead of a heading.

Render your document to generate an **HTML** file.

The previously created **HTML** file will be replaced with the newly generated one.

You now have headings and subheadings in your **Quarto** document.

Data Analysis

AUTHOR
Dr. Muhammad Saufi

PUBLISHED
August 14, 2024

Setting the Environment

Loading the Packages

```
# Load the packages into R session
library(tidyverse)
```

Warning: package 'tidyverse' was built under R version 4.4.1

— Attaching core tidyverse packages — tidyverse 2.0.0 —

| | | | |
|-------------|-------|-----------|-------|
| ✓ dplyr | 1.1.4 | ✓ readr | 2.1.5 |
| ✓ forcats | 1.0.0 | ✓ stringr | 1.5.1 |
| ✓ ggplot2 | 3.5.1 | ✓ tibble | 3.2.1 |
| ✓ lubridate | 1.9.3 | ✓ tidyr | 1.3.1 |
| ✓ purrr | 1.0.2 | | |

— Conflicts — tidyverse_conflicts() —

✗ dplyr::filter() masks stats::filter()
✗ dplyr::lag() masks stats::lag()

ℹ Use the conflicted package (<<http://conflicted.r-lib.org/>>) to force all conflicts to become errors

Dataset

Reading the Dataset

```
# Import the dataset into workspace
penguins <- read_csv("Datasets/penguins.csv")
```

1.4 Section Numbering

As your document grows longer, you might have several headings and subheadings to effectively organize your content.

Numbered headings can facilitate easier navigation of your output report.

Instead of manually typing the numbers into the headings, you can automatically create numbered headings by adding ``number-sections: true`` into the **YAML** header.

*Notice the change from the previous ``format: html``. This ensures that subsequent code is generated as **HTML** output.*

Ensure the position of each line of code is correct as shown above.

```
---
title: "Data Analysis"
author: "Dr. Muhammad Saufi"
date: 14 August 2024
format:
  html:
    number-sections: true
editor: visual
---
```

Data Analysis

AUTHOR
Dr. Muhammad Saufi

PUBLISHED
August 14, 2024

1 Setting the Environment

1.1 Loading the Packages

```
# Load the packages into R session
library(tidyverse)
```

Warning: package 'tidyverse' was built under R version 4.4.1

```
— Attaching core tidyverse packages — tidyverse 2.0.0 —
✓ dplyr      1.1.4    ✓ readr      2.1.5
✓ forcats    1.0.0    ✓ stringr    1.5.1
✓ ggplot2    3.5.1    ✓ tibble     3.2.1
✓ lubridate  1.9.3    ✓ tidyr      1.3.1
✓ purrr      1.0.2
— Conflicts — tidyverse_conflicts() —
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()     masks stats::lag()
! Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

2 Dataset

2.1 Reading the Dataset

```
# Import the dataset into workspace
penguins <- read_csv("Datasets/penguins.csv")
```

The numbered headings and subheadings appear in the **HTML** output after you render the document.

*Remember, you must render your document each time after you finish making changes in your report to generate a new **HTML** output.*

1.5 Table of Contents

You can automatically generate a table of contents in your output document by adjusting the **YAML** header. The following code demonstrates how to set up and customize the table of contents:

```
---
title: "Data Analysis"
author: "Dr. Muhammad Saufi"
date: 14 August 2024
format:
  html:
    toc: true
    toc-title: Contents
    toc-location: left
    toc-depth: 3
    toc-expand: 1
    number-sections: true
editor: visual
---
```

Below is an explanation of each line of code used to create the table of contents:

```
toc: true
```

Enables the table of contents

```
toc-title: Contents
```

Sets the title of the table of contents

```
toc-location: left
```

Positions the table of contents on the left side

```
toc-depth: 3
```

Specifies how many levels to be included

```
toc-expand: 1
```

Specifies how many levels to show initially

Below is the **HTML** output with the generated table of contents.

| | |
|---------------------------|----------------------------------|
| Contents | <h2>2 Dataset</h2> |
| 1 Setting the Environment | |
| 2 Dataset | |
| 2.1 Reading the Dataset | <h3>2.1 Reading the Dataset</h3> |
| 2.2 Exploring the Dataset | |

```
# Import the dataset into workspace
penguins <- read_csv("Datasets/penguins.csv")
```

Rows: 340 Columns: 9
— Column specification —
Delimiter: ","
chr (3): species, island, sex
dbl (6): rowid, bill_length_mm, bill_depth_mm, flipper_length_mm, body_mass_...

i Use ``spec()`` to retrieve the full column specification for this data.

i Specify the column types or set ``show_col_types = FALSE`` to quiet this message.

2.2 Exploring the Dataset

```
# Get a quick overview of dataset
glimpse(penguins)
```

1.6 Folding Code

Sometimes, code can take up a lot of space in your document, which might overwhelm readers who are more interested in the results rather than the code itself.

To keep your document clean and focused, you can use the ``code-fold`` option. This feature allows you to hide the code by default, with an option to reveal it using a button if necessary.

This way, readers can focus on the output while still having access to the code when needed.

```
---
title: "Data Analysis"
author: "Dr. Muhammad Saufi"
date: 14 August 2024
format:
  html:
    toc: true
    toc-title: Contents
    toc-location: left
    toc-depth: 3
    toc-expand: 1
    number-sections: true
    code-fold: true
    code-summary: "Show the Code"
editor: visual
---
```


Contents

1 Setting the Environment

1.1 Loading the Packages

2 Dataset

Data Analysis

AUTHOR

Dr. Muhammad Saufi

PUBLISHED

August 14, 2024

1 Setting the Environment

1.1 Loading the Packages

► Show the Code

Warning: package 'tidyverse' was built under R version 4.4.1

— Attaching core tidyverse packages — tidyverse 2.0.0 —

✓ dplyr 1.1.4 ✓ readr 2.1.5

✓ forcats 1.0.0 ✓ stringr 1.5.1

✓ ggplot2 3.5.1 ✓ tibble 3.2.1

✓ lubridate 1.9.3 ✓ tidyr 1.3.1

✓ purrr 1.0.2

— Conflicts — tidyverse_conflicts() —

✗ dplyr::filter() masks stats::filter()

✗ dplyr::lag() masks stats::lag()

ℹ Use the conflicted package (<<http://conflicted.r-lib.org/>>) to force all conflicts to be resolved

Readers can click the '**Show the Code**' button to view the hidden code when needed.

1.7 HTML Themes

Our eyes can sometimes get strained from looking at the monitor for too long. To reduce this strain, you can add a dark theme to your **HTML** output.

Even better, **HTML** allows you to switch between light and dark modes. You just need to specify this in the **YAML** header, as shown in the example below.

```
---
title: "Data Analysis"
author: "Dr. Muhammad Saufi"
date: 14 August 2024
format:
  html:
    toc: true
    toc-title: Contents
    toc-location: left
    toc-depth: 3
    toc-expand: 1
    number-sections: true
    code-fold: true
    code-summary: "Show the Code"
    theme:
      light: united
      dark: cyborg
editor: visual
---
```

Contents

1 Setting the Environment

1.1 Loading the Packages

2 Dataset

Data Analysis

AUTHOR

Dr. Muhammad Saufi

PUBLISHED

August 14, 2024

1 Setting the Environment

1.1 Loading the Packages

► Show the Code

Warning: package 'tidyverse' was built under R version 4.4.1

— Attaching core tidyverse packages ————— tid

✓ dplyr 1.1.4 ✓ readr 2.1.5

✓ forcats 1.0.0 ✓ stringr 1.5.1

✓ ggplot2 3.5.1 ✓ tibble 3.2.1

✓ lubridate 1.9.3 ✓ tidyr 1.3.1

✓ purrr 1.0.2

— Conflicts ————— tidyverse

✗ dplyr::filter() masks stats::filter()

✗ dplyr::lag() masks stats::lag()

i Use the conflicted package (<<http://conflicted.r-lib.org/>>) to ·



Click the button at the top right of the HTML file to enable or disable dark mode.

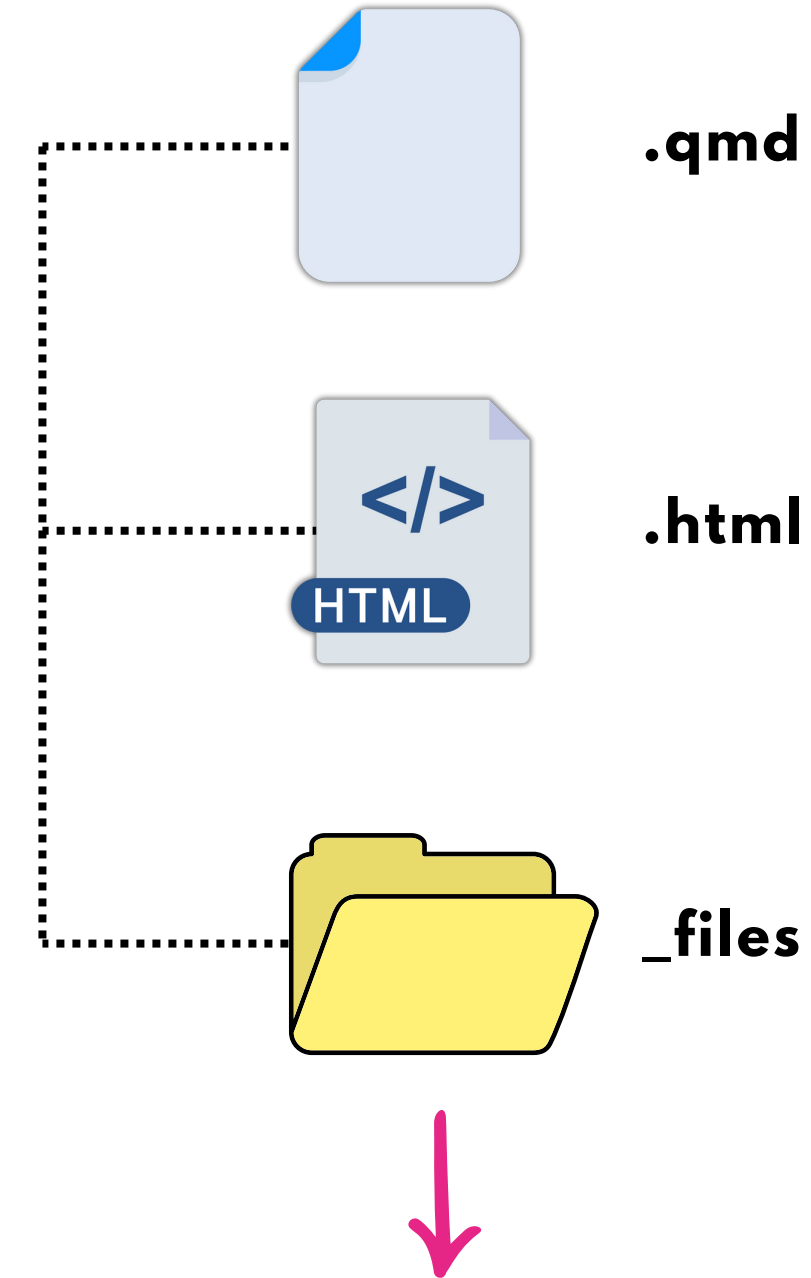
*You can explore a variety of themes provided in this [guide](#). **Quarto** offers at least 25 themes from the Bootswatch project.*

1.8 Self Contained

When you render a **Quarto** document into **HTML**, an additional folder will appear alongside the **HTML** file.

This folder typically contains assets like images and other resources referenced by the **HTML** file to ensure it displays correctly.

If you move or share the **HTML** file, you'll need to include this folder so that all associated content is correctly displayed.



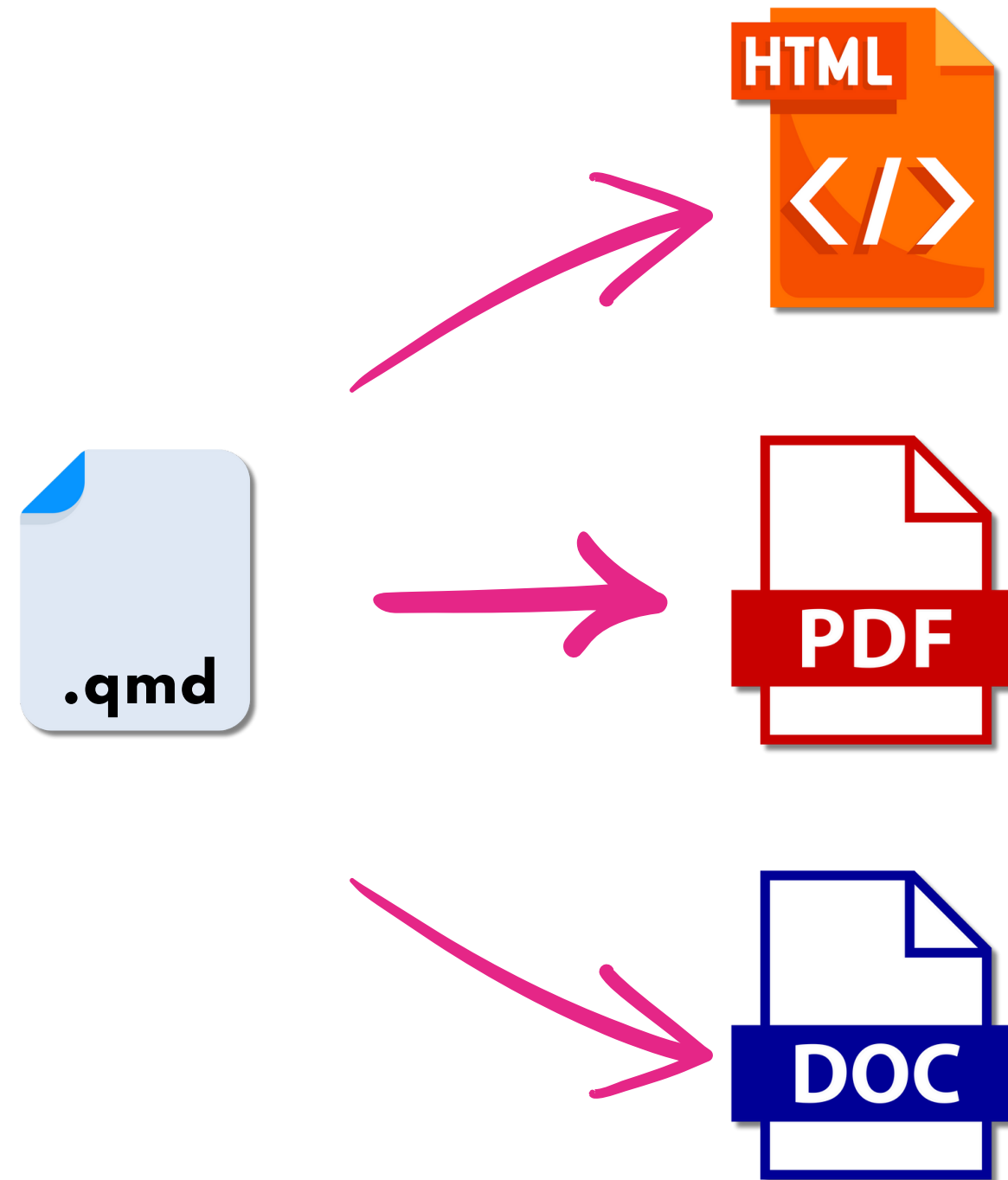
This folder contains external dependencies for the HTML file.

To create a fully self-contained **HTML** document, you can specify the ``embed-resources`` option in the **YAML** header.

```
---
title: "Data Analysis"
author: "Dr. Muhammad Saufi"
date: 14 August 2024
format:
  html:
    toc: true
    toc-title: Contents
    toc-location: left
    toc-depth: 3
    toc-expand: 1
    number-sections: true
    code-fold: true
    code-summary: "Show the Code"
    theme:
      light: united
      dark: cyborg
    embed-resources: true
editor: visual
---
```

*You should first delete the previous **HTML** document along with its external dependencies folder. Then, when you render the **Quarto** document again, it will generate only the **HTML** file.*

So far, we've focused on rendering **Quarto** documents into **HTML**. However, **Quarto** also allows you to generate **PDF** or **Microsoft Word** documents.

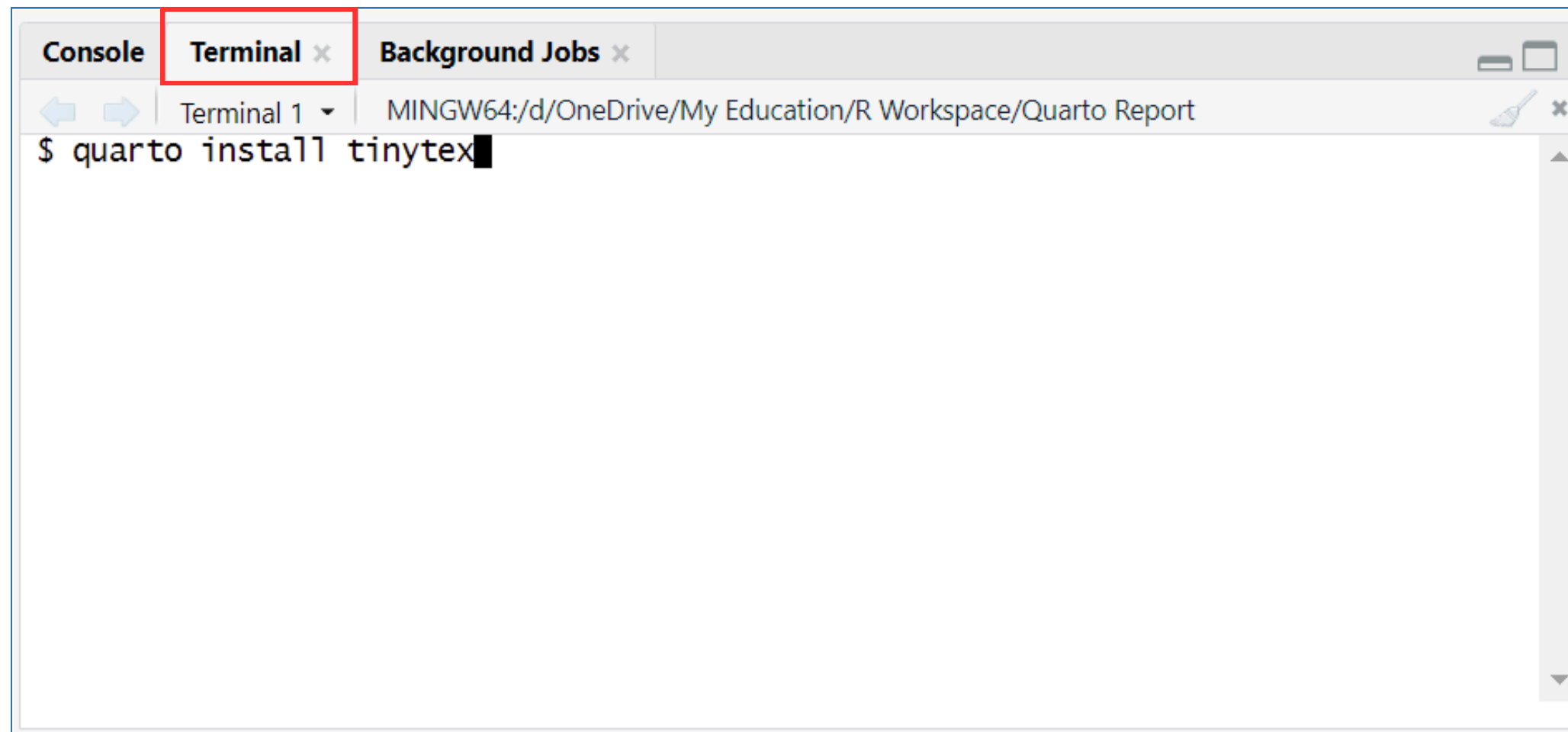


1.9 PDF

To create **PDF**, you'll need to install **TinyTeX** using the **Terminal** in **RStudio**. You can do this with the following command:

In Terminal:

```
quarto install tinytex
```



*Copy and paste the command into the **Terminal** and press **`Enter`**. The **TinyTeX** installation should start immediately.*

To create a **PDF** output, use the ``pdf`` format in the **YAML** header. The option ``toc: true`` will generate a table of contents in the **PDF**. If you want each heading numbered, include ``number-sections: true``.

```
---
title: "Data Analysis"
author: "Dr. Muhammad Saufi"
date: 14 August 2024
format:
  pdf:
    toc: true
    toc-title: Contents
    number-sections: true
editor: visual
---
```

*Once you've set these options, click **`Render`** to generate the **PDF** document. You should find the **PDF** within your working directory.*

1.10 Microsoft Word

To create a **Microsoft Word** output, use the ``docx`` format in the **YAML** header. For example:

```
---  
title: "Data Analysis"  
author: "Dr. Muhammad Saufi"  
date: 14 August 2024  
format:  
  docx:  
    toc: true  
    toc-title: Contents  
    number-sections: true  
editor: visual  
---
```

*Your **Microsoft Word** document will appear in your working directory after you render your **Quarto** document.*

As you've noticed, we render the **Quarto** document into each output format (**HTML**, **PDF**, **Microsoft Word**) individually, not simultaneously.

To create the desired output, you need to specify the format in the **YAML** header.

It's recommended to start by rendering your document into **HTML**, as it allows you to preview your report while making changes.

Once you're satisfied with the final version of your report, you can then render the document into other formats, such as **PDF** or **Microsoft Word**.





*Thank
You*