

# Urban Pedals: Unveiling Toronto's Bike Share Story

## Overview

This report contains the code and data for a data analysis project focused on exploring the ridership patterns of the bike-share scheme in Toronto. The analysis is conducted using the R programming language to gain insights into user behaviors, popular routes, and temporal trends.

---

## Data Source

- Ref:
    - <https://open.toronto.ca/dataset/bike-share-toronto-ridership-data/> (<https://open.toronto.ca/dataset/bike-share-toronto-ridership-data/>)
- 

## Download and Import data.

- Import libraries.

```
# Load library
library(opendatatoronto)
library(dplyr)
```

```
##
## 载入程辑包: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##   filter, lag
```

```
## The following objects are masked from 'package:base':
##
##   intersect, setdiff, setequal, union
```

```
library(tidyr)
library(ggplot2)
```

- Manually download data files from data source.
  - We have 52 files of 5-year data, from 2019 to 2023.
  - List of all data files

```
# Define path of dataset
```

```
DATA_PATH <- "./source/"
```

```
# get the paths for each data file.
```

```
FILE_PATH_LIST <- list.files(DATA_PATH, pattern = "\\*.csv$", full.names = TRUE)
```

```
FILE_PATH_LIST
```

```
## [1] "./source/2019-Q1.csv"
## [2] "./source/2019-Q2.csv"
## [3] "./source/2019-Q3.csv"
## [4] "./source/2019-Q4.csv"
## [5] "./source/2020-01.csv"
## [6] "./source/2020-02.csv"
## [7] "./source/2020-03.csv"
## [8] "./source/2020-04.csv"
## [9] "./source/2020-05.csv"
## [10] "./source/2020-06.csv"
## [11] "./source/2020-07.csv"
## [12] "./source/2020-08.csv"
## [13] "./source/2020-09.csv"
## [14] "./source/2020-10.csv"
## [15] "./source/2020-11.csv"
## [16] "./source/2020-12.csv"
## [17] "./source/Bike share ridership 2021-01.csv"
## [18] "./source/Bike share ridership 2021-02.csv"
## [19] "./source/Bike share ridership 2021-03.csv"
## [20] "./source/Bike share ridership 2021-04.csv"
## [21] "./source/Bike share ridership 2021-05.csv"
## [22] "./source/Bike share ridership 2021-06.csv"
## [23] "./source/Bike share ridership 2021-07.csv"
## [24] "./source/Bike share ridership 2021-08.csv"
## [25] "./source/Bike share ridership 2021-09.csv"
## [26] "./source/Bike share ridership 2021-10.csv"
## [27] "./source/Bike share ridership 2021-11.csv"
## [28] "./source/Bike share ridership 2021-12.csv"
## [29] "./source/Bike share ridership 2022-01.csv"
## [30] "./source/Bike share ridership 2022-02.csv"
## [31] "./source/Bike share ridership 2022-03.csv"
## [32] "./source/Bike share ridership 2022-04.csv"
## [33] "./source/Bike share ridership 2022-05.csv"
## [34] "./source/Bike share ridership 2022-06.csv"
## [35] "./source/Bike share ridership 2022-07.csv"
## [36] "./source/Bike share ridership 2022-08.csv"
## [37] "./source/Bike share ridership 2022-09.csv"
## [38] "./source/Bike share ridership 2022-10.csv"
## [39] "./source/Bike share ridership 2022-11.csv"
## [40] "./source/Bike share ridership 2022-12.csv"
## [41] "./source/Bike share ridership 2023-01.csv"
## [42] "./source/Bike share ridership 2023-02.csv"
```

```
## [43] "./source/Bike share ridership 2023-03.csv"
## [44] "./source/Bike share ridership 2023-04.csv"
## [45] "./source/Bike share ridership 2023-05.csv"
## [46] "./source/Bike share ridership 2023-06.csv"
## [47] "./source/Bike share ridership 2023-07.csv"
## [48] "./source/Bike share ridership 2023-08.csv"
## [49] "./source/Bike share ridership 2023-09.csv"
## [50] "./source/Bike share ridership 2023-10.csv"
## [51] "./source/Bike share ridership 2023-11.csv"
## [52] "./source/Bike share ridership 2023-12.csv"
```

- Import data from data files.
  - Check the columns of each csv file for data consistence.
  - Combine all data into a uniformed dataframe for further anaysis.

```
# # Apply read.csv, a function to import data from csv file, for each file.
# # Get a list of df
# df_list <- lapply(FILE_PATH_LIST, read.csv)
#
#
# # Union all df by row
# raw_df <- do.call(rbind, df_list)
# raw_df
```

- 
- Test using selective data

```
# paths <- c(  
#   "./source/2019-Q1.csv",  
#   "./source/2019-Q2.csv",  
#   "./source/2019-Q3.csv",  
#   "./source/2019-Q4.csv",  
#   "./source/2020-01.csv",  
#   "./source/2020-02.csv",  
#   "./source/2020-03.csv",  
#   "./source/2020-04.csv",  
#   "./source/2020-05.csv",  
#   "./source/2020-06.csv",  
#   "./source/2020-07.csv",  
#   "./source/2020-08.csv",  
#   "./source/2020-09.csv",  
#   "./source/2020-10.csv",  
#   "./source/2020-11.csv",  
#   "./source/2020-12.csv"  
# )  
# # paths <- c("./source/2020-01.csv")  
# df_list <- lapply(paths, read.csv)  
# raw_df <- do.call(rbind, df_list)  
# raw_df
```

---

## Data Processing

### Handling NA and NULL value

- Handle NA value that exist in the raw data

```
# # Remove rows with any NA values  
# proc_na_df <- raw_df[complete.cases(raw_df), ]  
#  
# # Remove rows with "NULL" values  
# proc_null_df <- proc_na_df %>%  
#   filter(!(End.Station.Name == "NULL" | End.Station.Id == "NULL" | Start.Station.Id == "NULL" | Start.Station.Name == "NULL"))  
# proc_df <- proc_null_df
```

# Converting Time-Dimension Data

- Divide time into year, month, date, hour, and minute.

```
# # Divide "Start.Time" into columns
# proc_df$Start.Time <- as.POSIXct(proc_df$Start.Time, format = "%m/%d/%Y %H:%M")
# proc_df$Start.Year <- as.factor(format(proc_df$Start.Time, "%Y"))
# proc_df$Start.Month <- as.factor(format(proc_df$Start.Time, "%m"))
# proc_df$Start.Date <- as.factor(format(proc_df$Start.Time, "%d"))
# proc_df$Start.Hours <- as.factor(format(proc_df$Start.Time, "%H"))
# proc_df$Start.Minutes <- as.factor(format(proc_df$Start.Time, "%M"))
#
# # Divide "End" into columns
# proc_df$End.Time <- as.POSIXct(proc_df$End.Time, format = "%m/%d/%Y %H:%M")
# proc_df$End.Year <- as.factor(format(proc_df$End.Time, "%Y"))
# proc_df$End.Month <- as.factor(format(proc_df$End.Time, "%m"))
# proc_df$End.Date <- as.factor(format(proc_df$End.Time, "%d"))
# proc_df$End.Hours <- as.factor(format(proc_df$End.Time, "%H"))
# proc_df$End.Minutes <- as.factor(format(proc_df$End.Time, "%M"))
#
# # factor user.type
# proc_df$User.Type <- as.factor(proc_df$User.Type)
#
#
# # Drop Start.Time and End.Time
# proc_df <- proc_df %>% select(-Start.Time, -End.Time)
```

- Check NA value again, in case of any possible values generated during the data processing.

```
# is_miss <- any(is.na(proc_df))
#
# # if the processed_df contains missing value, drop the rows with missing values and assign to df
# if (is_miss) {
#   df <- proc_df[complete.cases(proc_df), ]
# # otherwise, df = proc_df
# }else{
#   df <- proc_df
# }
#
# is_miss <- any(is.na(df))
# cat("df has missing value? ", is_miss) # output result
```

- Data Overview after data processing.

```
# # Data overview after data processing
# num_row <- nrow(df)           # total rows
# column_names <- colnames(df)  # column names
# cat("\n\nNumber of rows: ", "\n", num_row)
# cat("\n\nColumn names: ", "\n", column_names)
#
# cat("\n\nDisplay the Structure:\n")
# str(df)
#
# cat("\n\nDisplay Summaries:\n")
# summary(df)
```

- Preview data

```
# df
# head(df, 10)
```

---

## Exporting Processed Data(Optional)

- Export the Processed data to permanently store the processed data.

```
# # Path to export
# output_file <- "./data/dataset.csv"
# # export
# write.csv(df, file = output_file, row.names = FALSE)
```

---

## ReLoading Processed Data

```
data_file <- "./data/dataset.csv"
df <- read.csv(data_file)
df
```

- Data Overview afater loading data.

```
# Data overview after data processing
num_row <- nrow(df)          # total rows
column_names <- colnames(df) # column names
cat("\n\nNumber of rows: ", "\n", num_row)
```

```
##
##
## Number of rows:
## 17789237
```

```
cat("\n\nColumn names: ", "\n", column_names)
```

```
##
##
## Column names:
## Trip.Id Trip..Duration Start.Station.Id Start.Station.Name End.Station.Id End.Station.Name Bike.Id User.Type Start.Year
Start.Month Start.Date Start.Hours Start.Minutes End.Year End.Month End.Date End.Hours End.Minutes
```

```
cat("\n\nDisplay the Structure:\n")
```

```
##
##
## Display the Structure:
```

```
str(df)
```



```
## 'data.frame': 17789237 obs. of 18 variables:
## $ Trip.Id : num 4581278 4581279 4581280 4581281 4581282 ...
## $ Trip..Duration : int 1547 1112 589 259 281 624 604 416 192 518 ...
## $ Start.Station.Id : int 7021 7160 7055 7012 7041 7041 7041 7275 7071 7199 ...
## $ Start.Station.Name: chr "Bay St / Albert St" "King St W / Tecumseth St" "Jarvis St / Carlton St" "Elizabeth St / Edwa
rd St (Bus Terminal)" ...
## $ End.Station.Id : int 7233 7051 7013 7235 7257 7031 7031 7041 7311 7252 ...
## $ End.Station.Name : chr "King / Cowan Ave - SMART" "Wellesley St E / Yonge St (Green P)" "Scott St / The Esplanade"
"Bay St / College St (West Side) - SMART" ...
## $ Bike.Id : chr "1296" "2947" "2293" "283" ...
## $ User.Type : chr "Annual Member" "Annual Member" "Annual Member" "Annual Member" ...
## $ Start.Year : int 2019 2019 2019 2019 2019 2019 2019 2019 2019 2019 ...
## $ Start.Month : int 1 1 1 1 1 1 1 1 1 1 ...
## $ Start.Date : int 1 1 1 1 1 1 1 1 1 1 ...
## $ Start.Hours : int 0 0 0 0 0 0 0 0 0 0 ...
## $ Start.Minutes : int 8 10 15 16 19 26 26 26 34 38 ...
## $ End.Year : int 2019 2019 2019 2019 2019 2019 2019 2019 2019 2019 ...
## $ End.Month : int 1 1 1 1 1 1 1 1 1 1 ...
## $ End.Date : int 1 1 1 1 1 1 1 1 1 1 ...
## $ End.Hours : int 0 0 0 0 0 0 0 0 0 0 ...
## $ End.Minutes : int 33 29 25 20 24 36 36 33 37 46 ...
```

```
cat("\n\nDisplay Summaries:\n")
```

```
##
##
## Display Summaries:
```

```
summary(df)
```

```
##      Trip.Id      Trip..Duration      Start.Station.Id Start.Station.Name
## Min.      : 4581278 Min.      :      0 Min.      :7000      Length:17789237
## 1st Qu.: 9607762 1st Qu.:      439 1st Qu.:7078      Class :character
## Median :14783657 Median :      729 Median :7227      Mode  :character
## Mean    :15075490 Mean     :    1047 Mean     :7246
## 3rd Qu.:20359050 3rd Qu.:    1164 3rd Qu.:7383
## Max.    :26682738 Max.     :12403785 Max.     :7681
## End.Station.Id End.Station.Name      Bike.Id      User.Type
## Min.      :7000 Length:17789237 Length:17789237 Length:17789237
## 1st Qu.:7077 Class :character Class :character Class :character
## Median :7224 Mode  :character Mode  :character Mode  :character
## Mean     :7243
## 3rd Qu.:7381
## Max.     :7681
##      Start.Year      Start.Month      Start.Date      Start.Hours
## Min.      :2019 Min.      : 1.000 Min.      : 1.00 Min.      : 0.00
## 1st Qu.:2020 1st Qu.: 6.000 1st Qu.: 8.00 1st Qu.:11.00
## Median :2021 Median : 7.000 Median :16.00 Median :15.00
## Mean     :2021 Mean     : 7.246 Mean     :15.73 Mean     :14.57
## 3rd Qu.:2023 3rd Qu.: 9.000 3rd Qu.:23.00 3rd Qu.:18.00
## Max.     :2023 Max.     :12.000 Max.     :31.00 Max.     :23.00
## Start.Minutes      End.Year      End.Month      End.Date
## Min.      : 0.00 Min.      :2019 Min.      : 1.000 Min.      : 1.00
## 1st Qu.:14.00 1st Qu.:2020 1st Qu.: 6.000 1st Qu.: 8.00
## Median :30.00 Median :2021 Median : 7.000 Median :16.00
## Mean     :29.55 Mean     :2021 Mean     : 7.246 Mean     :15.73
## 3rd Qu.:45.00 3rd Qu.:2023 3rd Qu.: 9.000 3rd Qu.:23.00
## Max.     :59.00 Max.     :2024 Max.     :12.000 Max.     :31.00
##      End.Hours      End.Minutes
## Min.      : 0.00 Min.      : 0.00
## 1st Qu.:11.00 1st Qu.:15.00
## Median :16.00 Median :30.00
## Mean     :14.71 Mean     :29.79
## 3rd Qu.:18.00 3rd Qu.:45.00
## Max.     :23.00 Max.     :59.00
```

- Preview data

```
# df
head(df, 10)
```

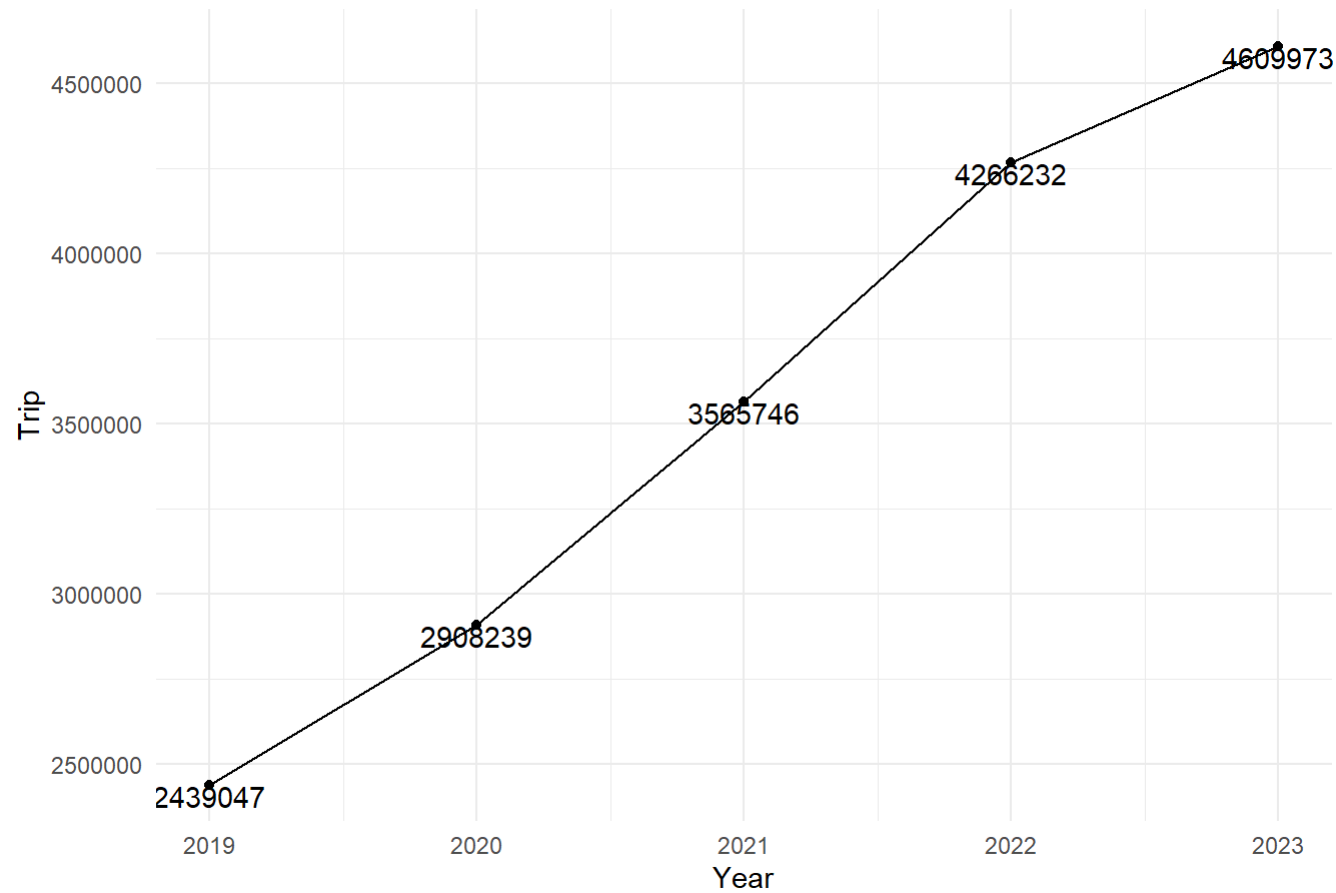
# Trip analysis

## Yearly Trip Trends

```
trip_yearly <- df %>%  
  group_by(Year = Start.Year) %>%  
  summarize(Trip = n())  
trip_yearly
```

```
ggplot(  
  data = trip_yearly,  
  mapping = aes(  
    x = Year,  
    y = Trip,  
    group = 1  
  )  
) +  
  geom_line() +  
  geom_point() +  
  geom_text(aes(label = as.character(Trip)), vjust = 1, hjust = 0.5) +  
  labs(  
    title = "Yearly Trip Trend",  
    x = "Year",  
    y = "Trip"  
  ) +  
  theme_minimal()
```

## Yearly Trip Trend



## Monthly Trip Distribution

- Comparing monthly trip to unveil patterns over the months

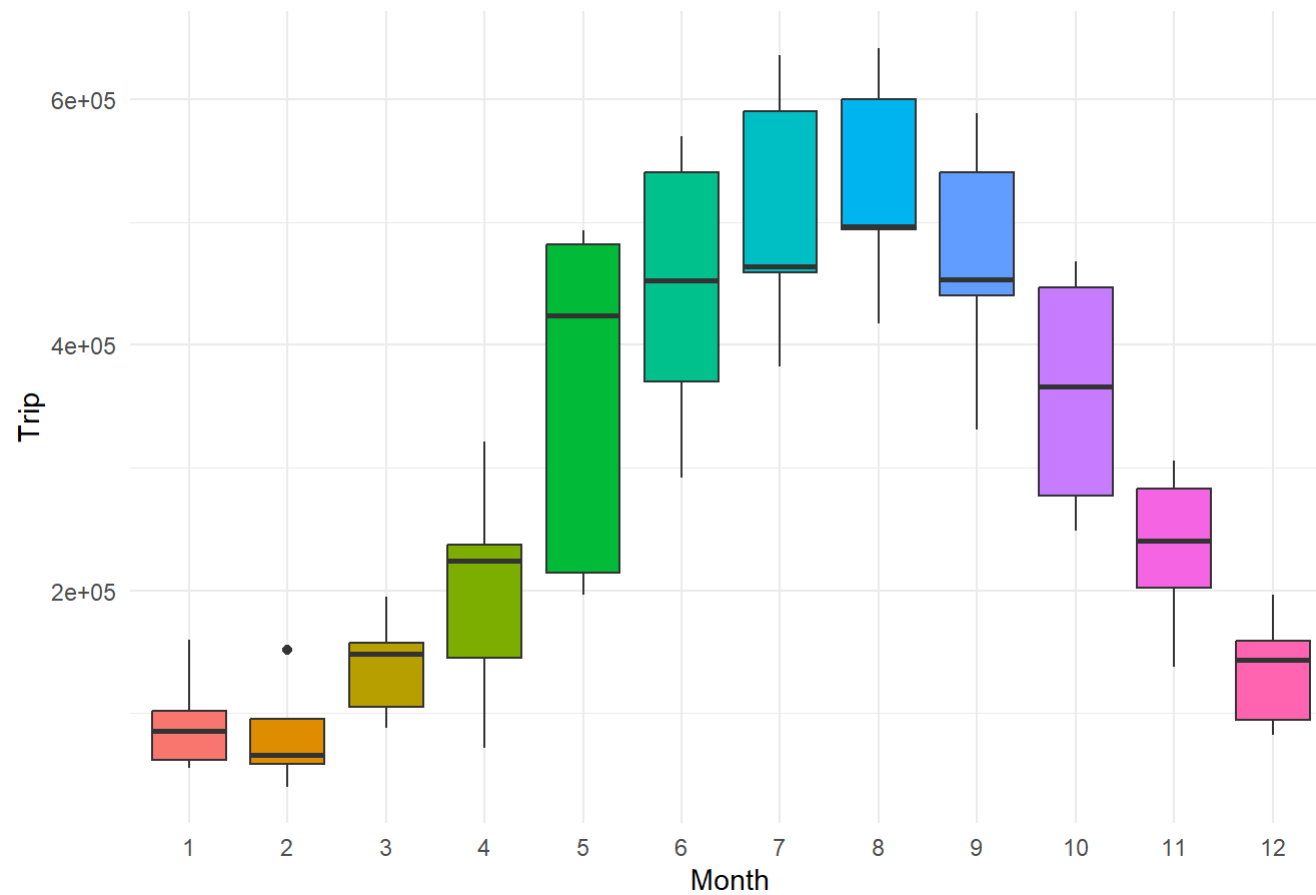
```
trip_monthly_across_year <- df %>%  
  group_by(  
    Year = Start.Year,  
    Month = Start.Month  
  ) %>%  
  summarize(Trip = n())
```

```
## `summarise()` has grouped output by 'Year'. You can override using the  
## `.groups` argument.
```

trip\_monthly\_across\_year

```
ggplot(  
  data = trip_monthly_across_year,  
  mapping = aes(  
    x = factor(Month),  
    y = Trip,  
    fill = factor(Month)  
  )  
) +  
geom_boxplot(show.legend = FALSE) +  
labs(  
  title = "Monthly Distribution of Trips",  
  x = "Month",  
  y = "Trip",  
  fill = "Month"  
) +  
theme_minimal()
```

## Monthly Distribution of Trips



## Hourly Trip Pattern

- Exploring Patterns Throughout the Day

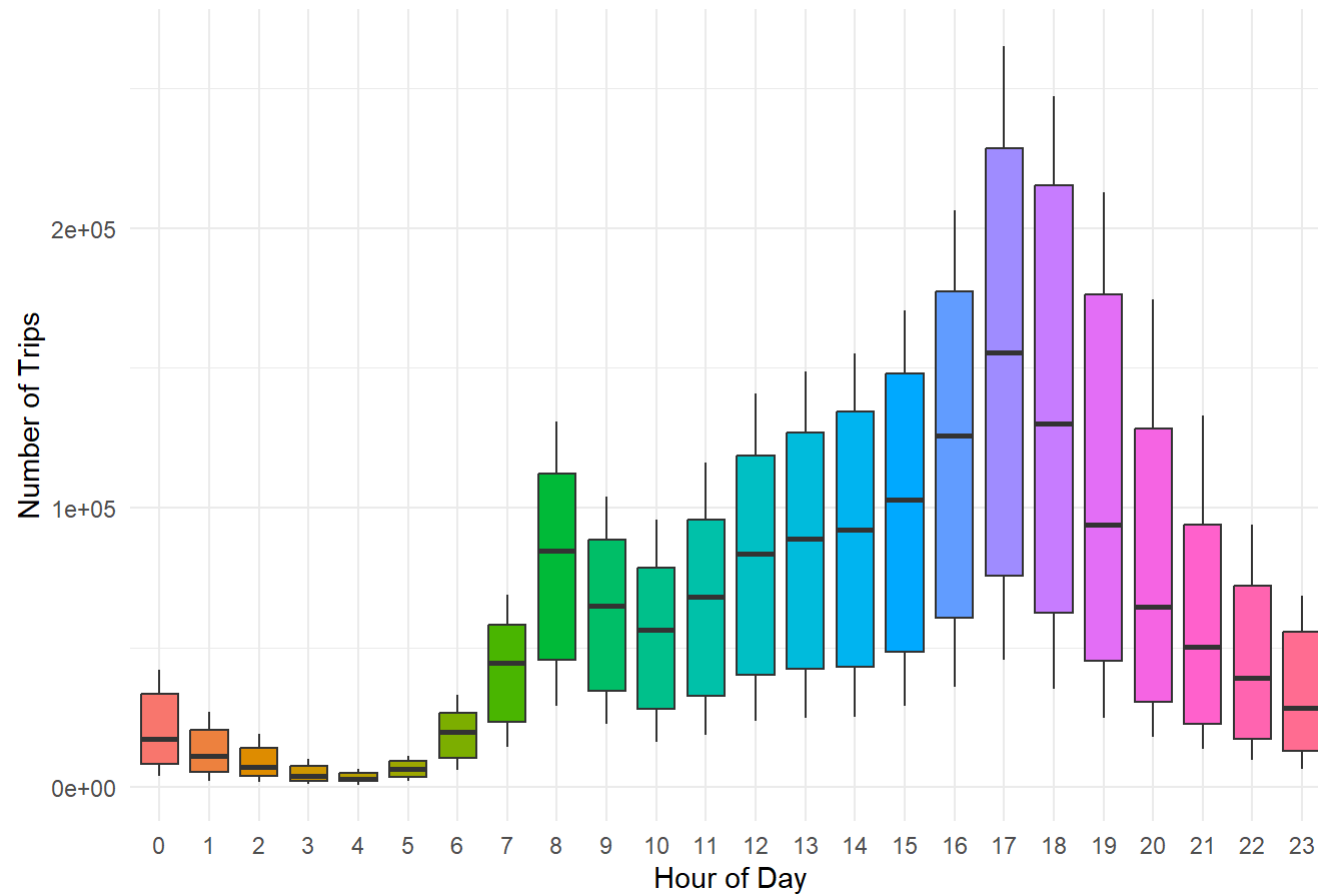
```
trip_hourly_across_month <- df %>%  
  group_by(  
    Month = Start.Month,  
    Hour = Start.Hours  
  ) %>%  
  summarize(Trip = n())
```

```
## `summarise()` has grouped output by 'Month'. You can override using the  
## `.groups` argument.
```

trip\_hourly\_across\_month

```
ggplot(  
  trip_hourly_across_month,  
  aes(  
    x = factor(Hour),  
    y = Trip,  
    fill = factor(Hour)  
  )  
) +  
geom_boxplot(show.legend = FALSE) +  
labs(  
  title = "Hourly Trips Distribution",  
  x = "Hour of Day",  
  y = "Number of Trips",  
  fill = "Hour of Day") +  
theme_minimal()
```

## Hourly Trips Distribution



## Duration Analysis

### Duration Yearly Trend

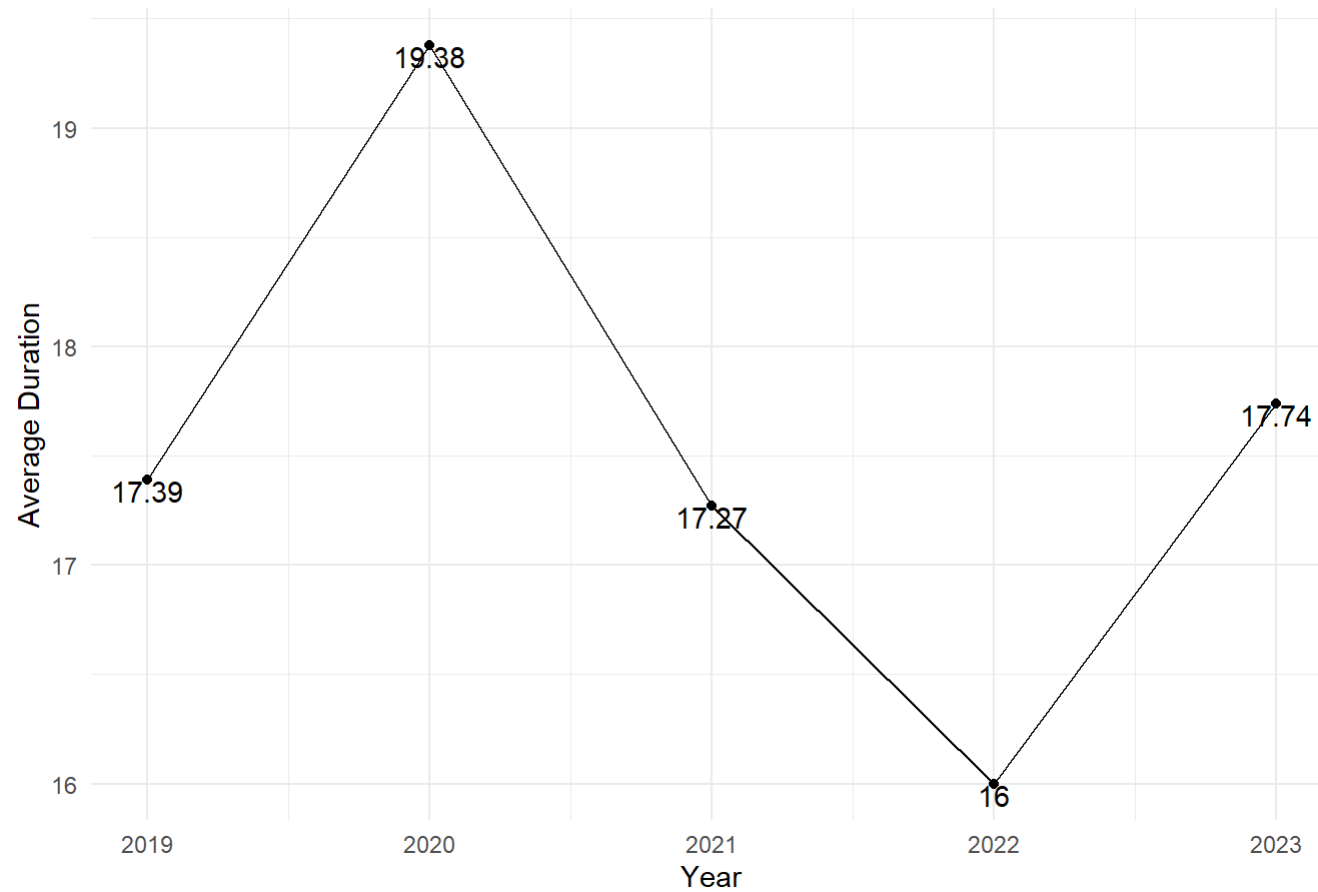
```
duration_yearly <- df %>%  
  group_by(Year = Start.Year) %>%  
  summarize(Mean_Duration = round(mean(Trip..Duration) / 60, 2))
```

```
duration_yearly
```



```
ggplot(  
  data = duration_yearly,  
  mapping = aes(  
    x = Year,  
    y = Mean_Duration,  
    group = 1  
  )  
) +  
  geom_line() +  
  geom_point() +  
  geom_text(  
    aes(  
      label = as.character(Mean_Duration)  
    ),  
    vjust = 1,  
    hjust = 0.5  
  ) +  
  labs(  
    title = "Average Duration Trend",  
    x = "Year",  
    y = "Average Duration"  
  ) +  
  theme_minimal()
```

## Average Duration Trend



## Duration Pattern over Months

```
duration_monthly_across_year <- df %>%  
  group_by(  
    Year = Start.Year,  
    Month = Start.Month) %>%  
  summarize(Mean_Duration = round(mean(Trip..Duration) / 60, 2)) # Convert mean to minutes
```

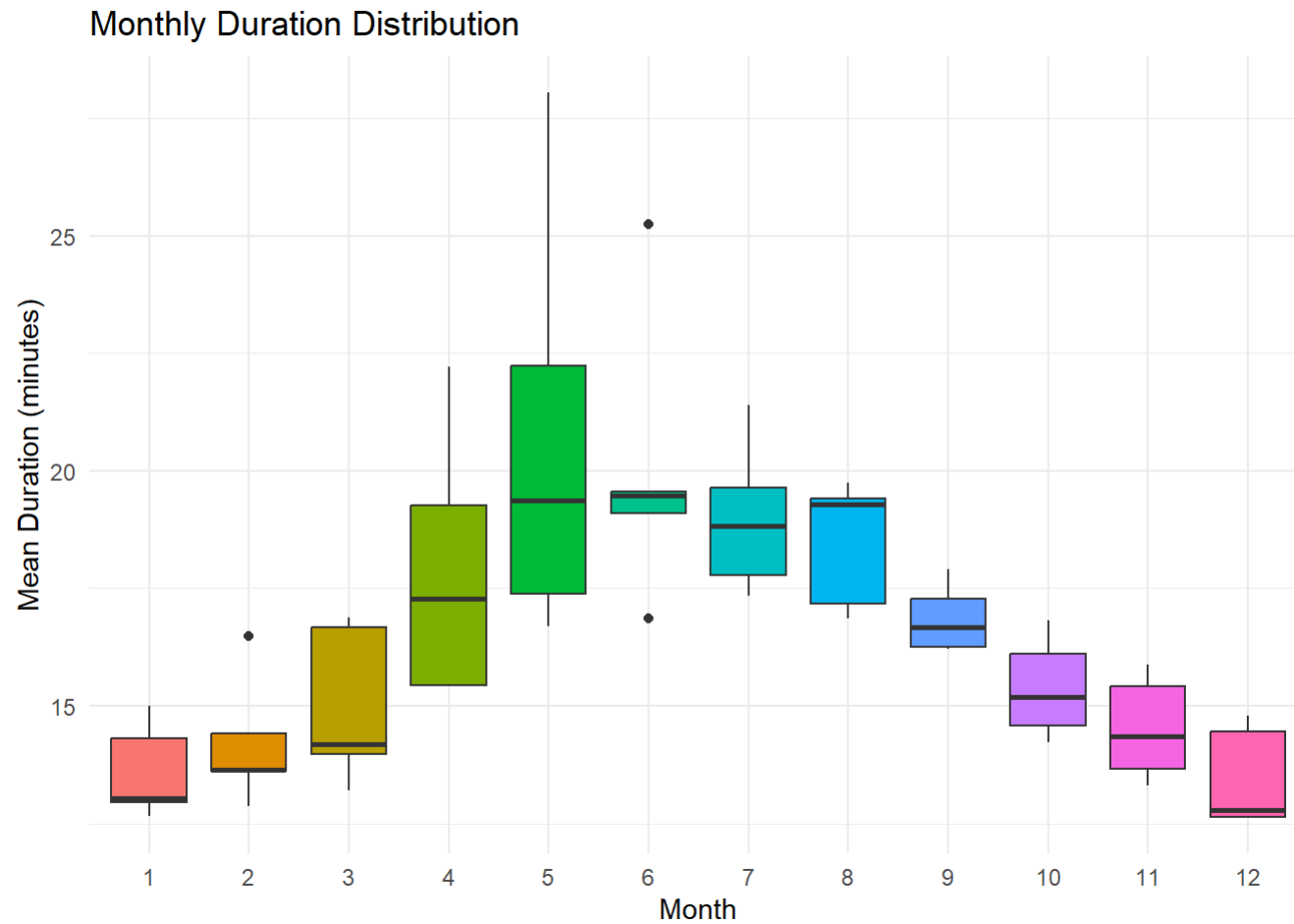
```
## `summarise()` has grouped output by 'Year'. You can override using the  
## `.groups` argument.
```

```
duration_monthly_across_year
```

```

ggplot(
  data = duration_monthly_across_year,
  mapping = aes(
    x = factor(Month),
    y = Mean_Duration,
    fill = factor(Month)
  )
) +
  geom_boxplot(show.legend = FALSE) +
  labs(
    title = "Monthly Duration Distribution",
    x = "Month",
    y = "Mean Duration (minutes)"
  ) +
  theme_minimal()

```



# Duration Pattern throughout A Day

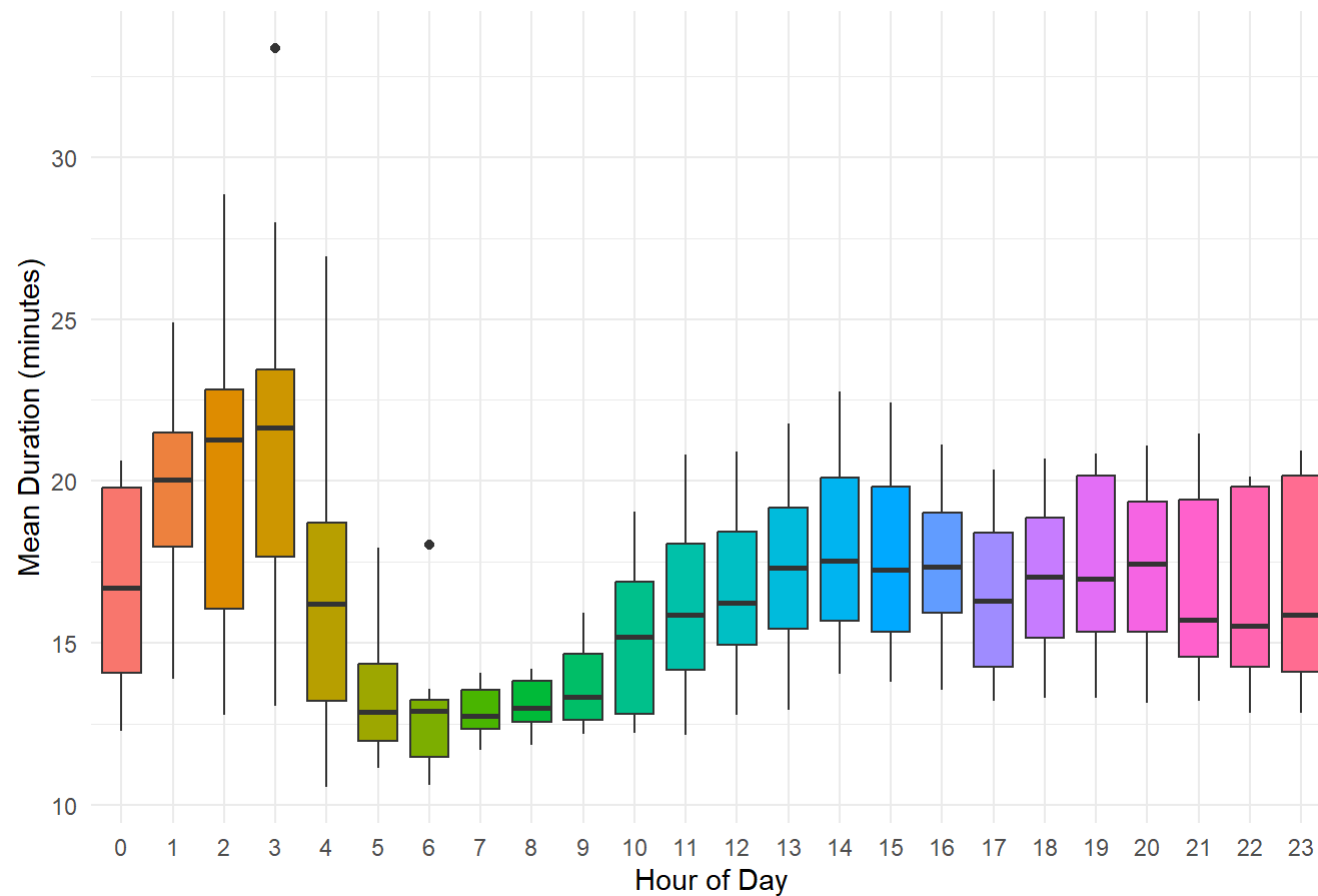
```
duration_hourly_across_month <- df %>%
  group_by(
    Month = Start.Month,
    Hour = Start.Hours
  ) %>%
  summarize(Mean_Duration = round(mean(Trip..Duration) / 60, 2)) # Convert mean to minutes
```

```
## `summarise()` has grouped output by 'Month'. You can override using the
## `.groups` argument.
```

```
duration_hourly_across_month
```

```
ggplot(
  data = duration_hourly_across_month,
  mapping = aes(
    x = factor(Hour),
    y = Mean_Duration,
    fill = factor(Hour)
  )
) +
  geom_boxplot(show.legend = FALSE) +
  labs(
    title = "Hourly Duration Distribution",
    x = "Hour of Day",
    y = "Mean Duration (minutes)"
  ) +
  theme_minimal()
```

## Hourly Duration Distribution



## Geolocation Analysis

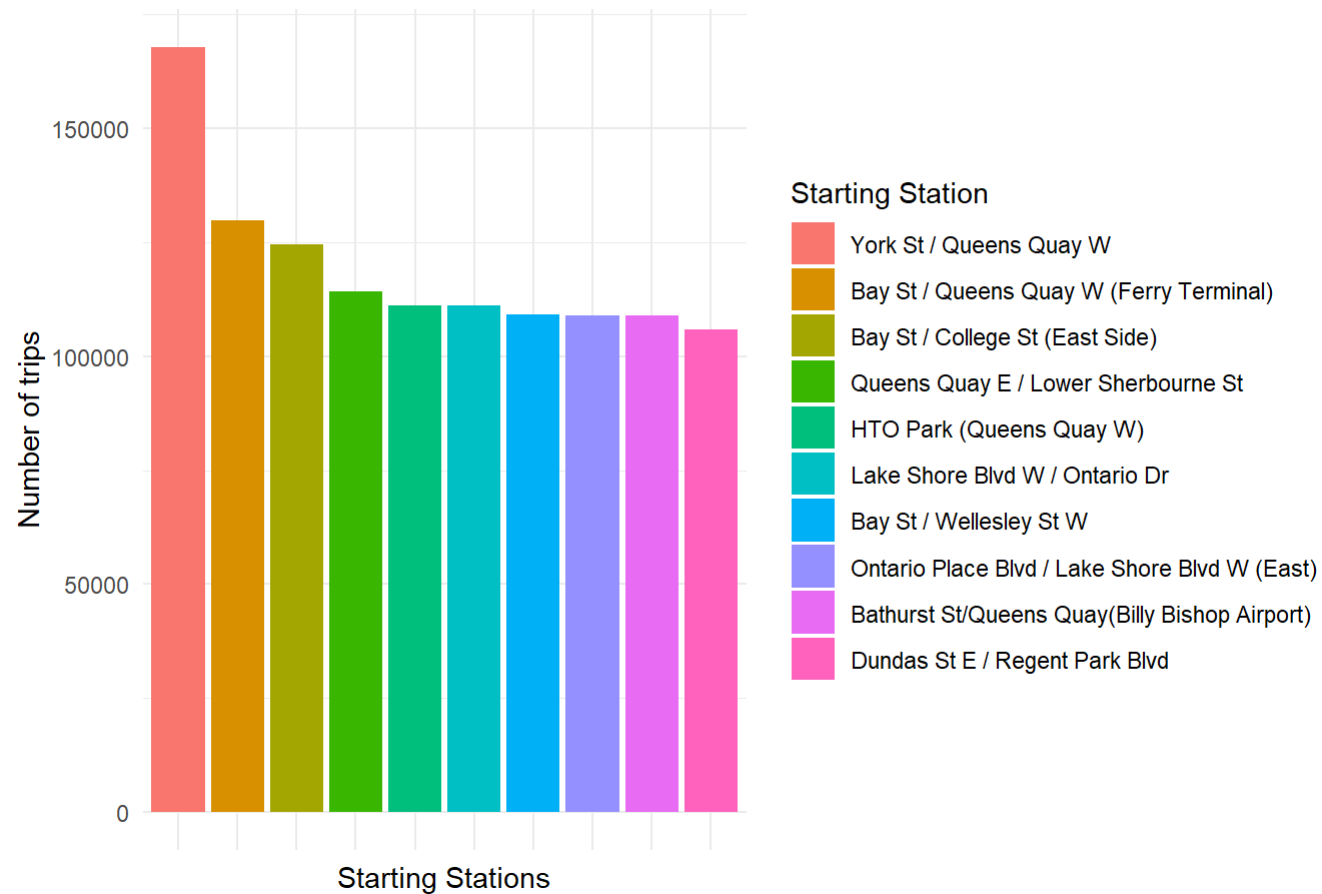
### Hot Start Locations

Hot start spot (Top 10)

```
hot_start_spot <- df %>%
  group_by(Start.Station.Name) %>%
  summarize(Total_Trips = n()) %>%
  arrange(desc(Total_Trips)) %>%
  slice_head(n = 10)
hot_start_spot
```

```
ggplot(  
  data = hot_start_spot,  
  mapping = aes(  
    x = reorder(Start.Station.Name, -Total_Trips),  
    y = Total_Trips,  
    fill = reorder(Start.Station.Name, -Total_Trips)  
  )  
) +  
geom_bar(  
  stat = "identity"  
) +  
labs(  
  title = "Total Number of trips per starting stations",  
  x = "Starting Stations",  
  y = "Number of trips",  
  fill = "Starting Station"  
) +  
theme_minimal()+  
theme(axis.text.x = element_blank())
```

Total Number of trips per starting stations



## Hot start street (Top 10)

```
library(stringr)
```

```
hot_start_streets <- df %>%
  separate_rows(Start.Station.Name, sep = " / ") %>% # Separate the station names
  mutate(Start.Station.Street = Start.Station.Name) %>%
  group_by(Start.Station.Street) %>%
  summarize(Total_Trips = n()) %>%
  arrange(desc(Total_Trips)) %>%
  slice_head(n = 10)
```

```
## Warning in gregexpr(pattern, x, perl = TRUE): 输入的字符串5347586不是UTF-8
```

```
## Warning in gregexpr(pattern, x, perl = TRUE): 输入的字符串5347588不是UTF-8
```

```
## Warning in gregexpr(pattern, x, perl = TRUE): 输入的字符串5350238不是UTF-8
```

```
## Warning in gregexpr(pattern, x, perl = TRUE): 输入的字符串5350495不是UTF-8
```

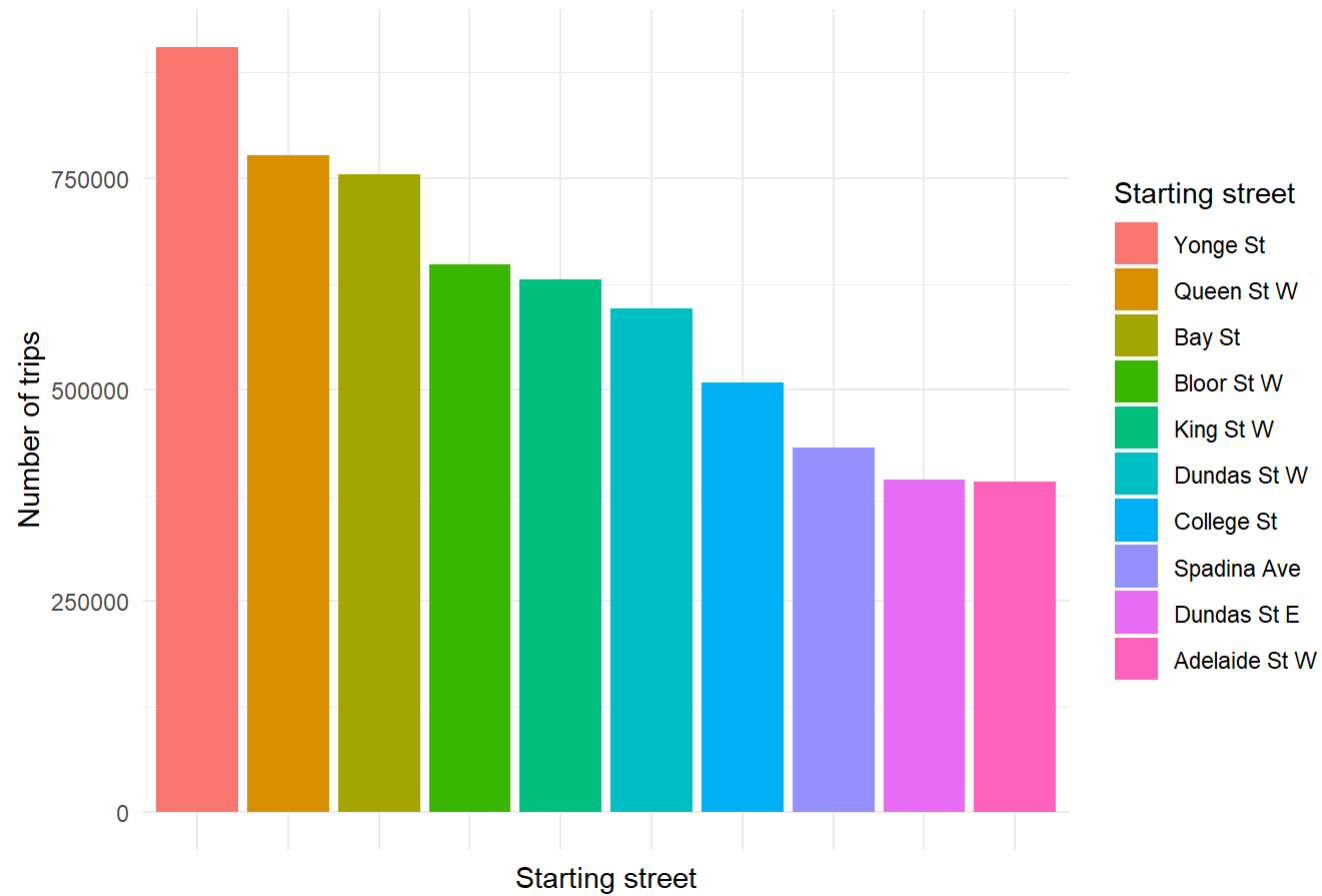
```
## Warning in gregexpr(pattern, x, perl = TRUE): 输入的字符串5350926不是UTF-8
```

```
hot_start_streets
```

```
ggplot(  
  data = hot_start_streets,  
  mapping = aes(  
    x = reorder(Start.Station.Street, -Total_Trips),  
    y = Total_Trips,  
    fill = reorder(Start.Station.Street, -Total_Trips)  
  )  
)+  
geom_bar(  
  stat = "identity"  
)+  
labs(  
  title = "Total Number of trips per starting street",  
  x = "Starting street",  
  y = "Number of trips",  
  fill = "Starting street"  
)+  
theme_minimal()+  
theme(axis.text.x = element_blank())
```



Total Number of trips per starting street

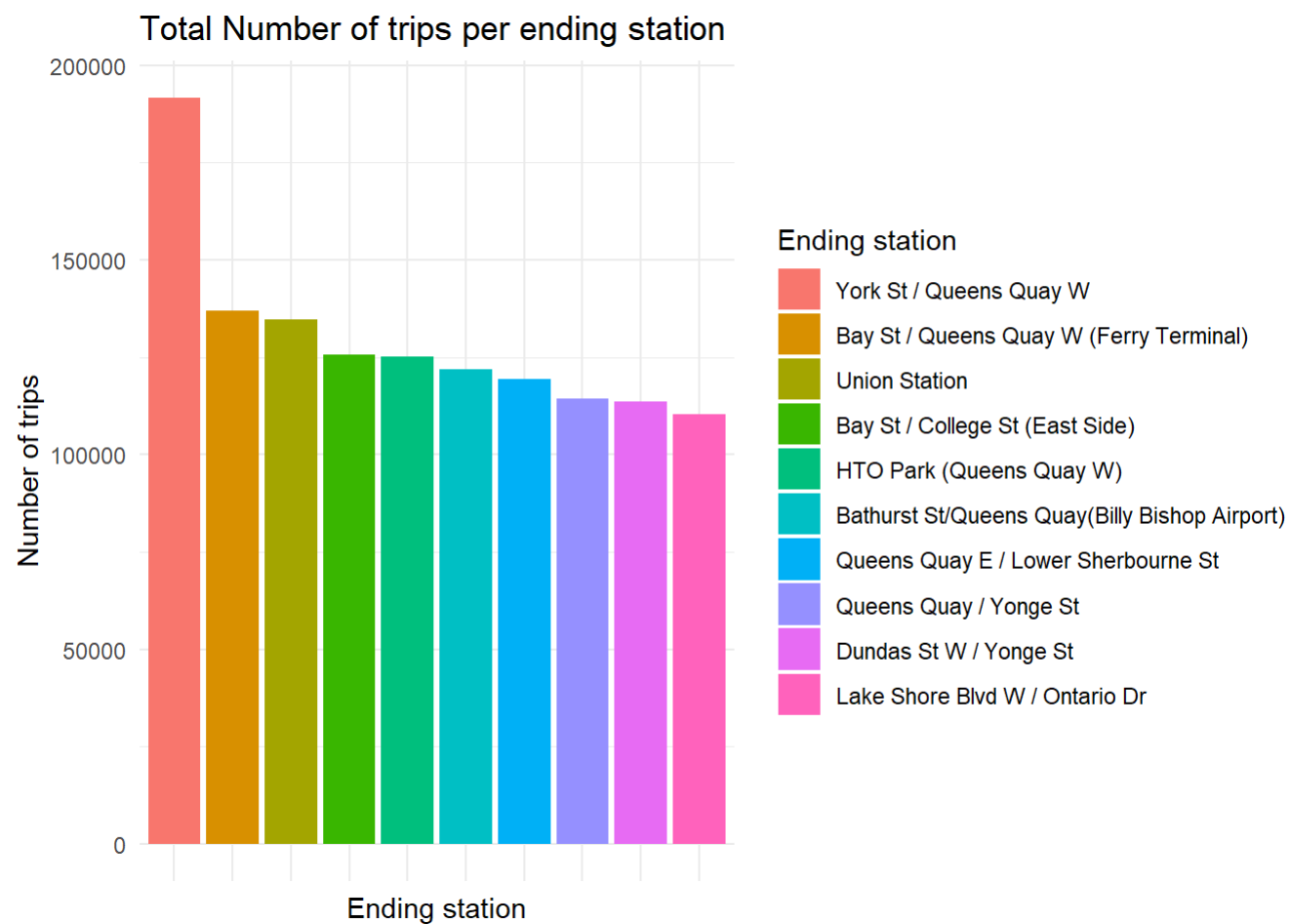


## Hot End Locations

### Hot End spot (Top 10)

```
hot_end_spot <- df %>%
  group_by(End.Station.Name) %>%
  summarize(Total_Trips = n()) %>%
  arrange(desc(Total_Trips)) %>%
  slice_head(n = 10)
hot_end_spot
```

```
ggplot(  
  data = hot_end_spot,  
  mapping = aes(  
    x = reorder(End.Station.Name, -Total_Trips),  
    y = Total_Trips,  
    fill = reorder(End.Station.Name, -Total_Trips)  
  )  
) +  
geom_bar(  
  stat = "identity"  
) +  
labs(  
  title = "Total Number of trips per ending station",  
  x = "Ending station",  
  y = "Number of trips",  
  fill = "Ending station"  
) +  
theme_minimal()+  
theme(axis.text.x = element_blank())
```



### Hot end street (Top 10)

```
hot_end_streets <- df %>%
  separate_rows(End.Station.Name, sep = " / ") %>% # Separate the station names
  mutate(End.Station.Street = End.Station.Name) %>%
  group_by(End.Station.Street) %>%
  summarize(Total_Trips = n()) %>%
  arrange(desc(Total_Trips)) %>%
  slice_head(n = 10)
```

```
## Warning in gregexpr(pattern, x, perl = TRUE): 输入的字符串5348071不是UTF-8
```

```
## Warning in gregexpr(pattern, x, perl = TRUE): 输入的字符串5348390不是UTF-8
```

```
## Warning in gregexpr(pattern, x, perl = TRUE): 输入的字符串5348452不是UTF-8
```

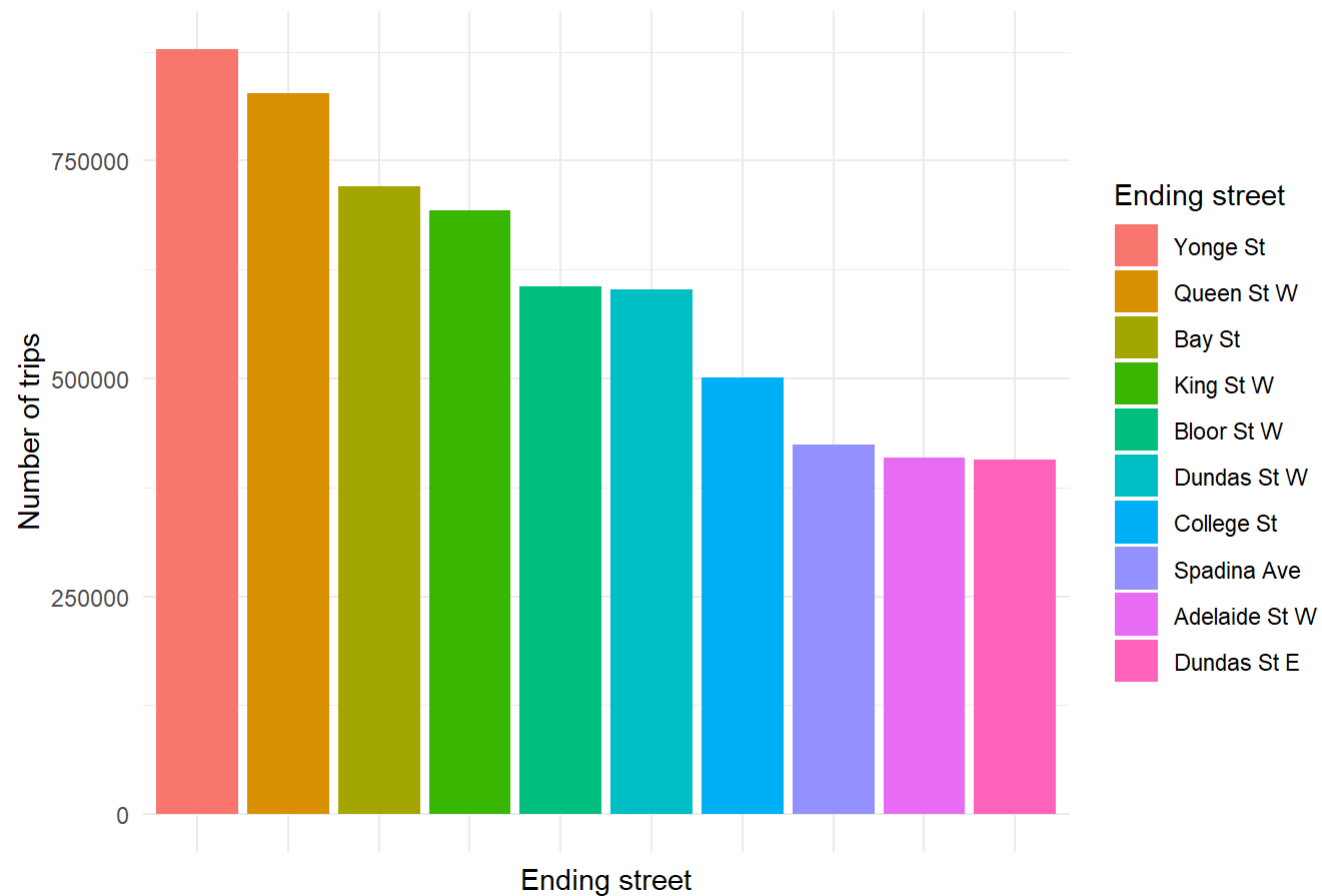
```
## Warning in gregexpr(pattern, x, perl = TRUE): 输入的字符串5349390不是UTF-8
```

```
## Warning in gregexpr(pattern, x, perl = TRUE): 输入的字符串5349530不是UTF-8
```

```
hot_end_streets
```

```
ggplot(  
  data = hot_end_streets,  
  mapping = aes(  
    x = reorder(End.Station.Street, -Total_Trips),  
    y = Total_Trips,  
    fill = reorder(End.Station.Street, -Total_Trips)  
  )  
) +  
geom_bar(  
  stat = "identity"  
) +  
labs(  
  title = "Total Number of trips per ending street",  
  x = "Ending street",  
  y = "Number of trips",  
  fill = "Ending street"  
) +  
theme_minimal()+  
theme(axis.text.x = element_blank())
```

Total Number of trips per ending street

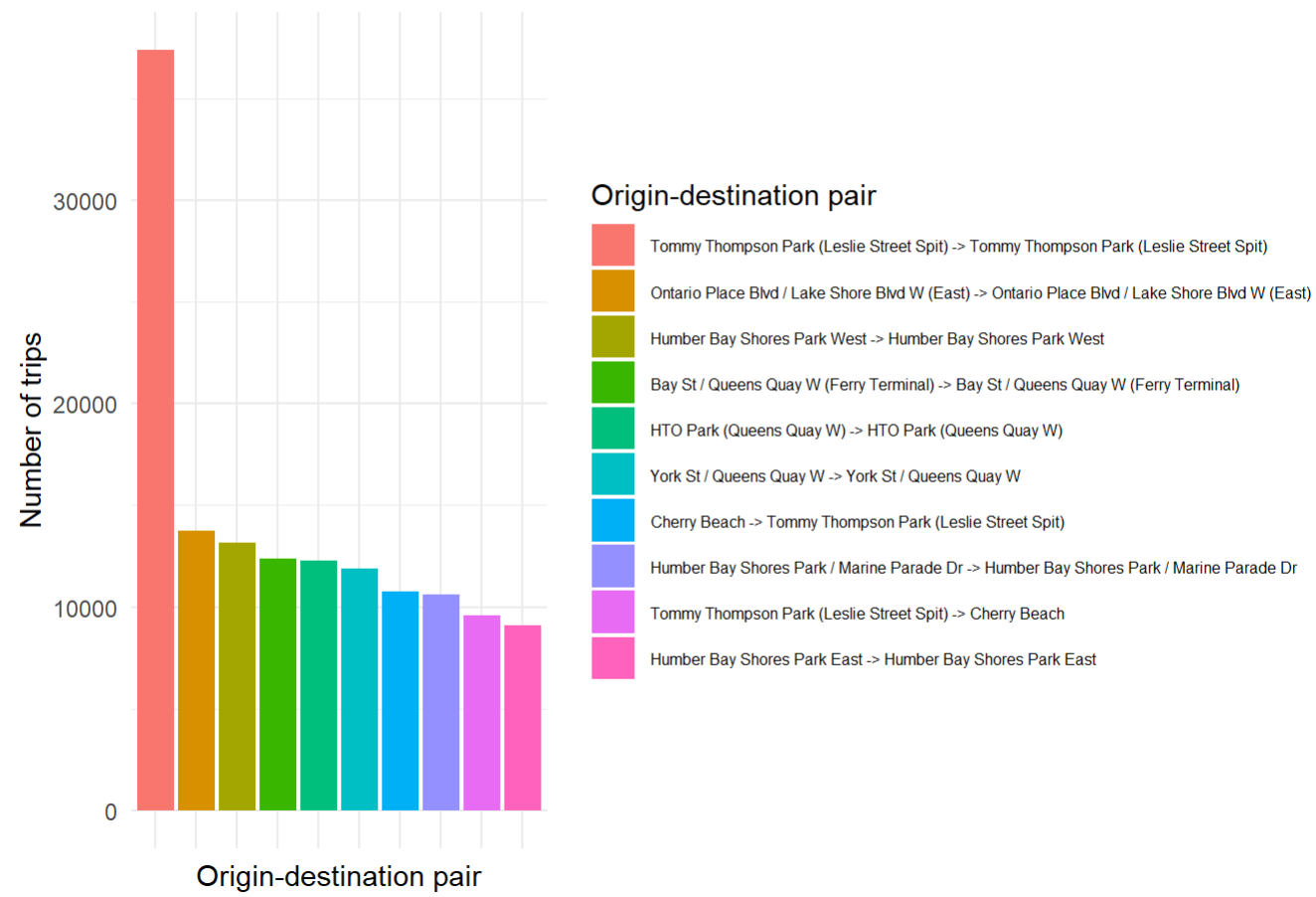


## Hot Trips (Top 10)

```
hot_trip <- df %>%
  mutate(Trip.Spots = paste(Start.Station.Name, End.Station.Name, sep = " -> ")) %>%
  group_by(Trip.Spots) %>%
  summarize(Total_Trips = n()) %>%
  arrange(desc(Total_Trips)) %>%
  slice_head(n = 10)
hot_trip
```

```
ggplot(  
  data = hot_trip,  
  mapping = aes(  
    x = reorder(Trip.Spots, -Total_Trips),  
    y = Total_Trips,  
    fill = reorder(Trip.Spots, -Total_Trips)  
  )  
) +  
geom_bar(  
  stat = "identity"  
) +  
labs(  
  title = "Total Number of trips per origin-destination pair",  
  x = "Origin-destination pair",  
  y = "Number of trips",  
  fill = "Origin-destination pair"  
) +  
theme_minimal()+  
theme(axis.text.x = element_blank() ,  
      legend.text = element_text(size = 6))
```

Total Number of trips per origin-destination pair



# Shiny Rendering

```
library(shiny)
library(ggplot2)

ui <- fluidPage(
  tags$head(
    tags$style(HTML("
      #sidebar {
        position: fixed;
        top: 0;
        left: 0;
        bottom: 0;
        width: 300px; /* Set your desired width */
        background-color: #f8f9fa; /* Set your desired background color */
        padding: 10px;

      },
      #mainPanel {
        font-size: 24px;
        padding-right: 200px;
      }
    "))
  ),

  sidebarLayout(
    sidebarPanel(
      id = "sidebar",
      tags$div(
        tags$h3(tags$a(href = "#title", "Urban Pedals")),
        tags$ul(
          tags$li(
            tags$a(href = "#source", "Data Source")
          ),
          tags$li(
            tags$a(href = "#processing", "Data Processing")
          ),
          # Trip
          tags$li(
            tags$a(href = "#trip", "Trip Analysis"),
            tags$ul(
```



```

tags$li(tags$a(href = "#yearly-trip", "Yearly Trends")),
tags$li(tags$a(href = "#monthly-trip", "The Pattern over Months")),
tags$li(tags$a(href = "#hourly-trip", "The Pattern throughout A Day"))
)
),
# duration
tags$li(tags$a(href = "#duration", "Duration Analysis"),
tags$ul(
tags$li(tags$a(href = "#yearly-duration", "Yearly Trend")),
tags$li(tags$a(href = "#monthly-duration", "The Pattern over Months")),
tags$li(tags$a(href = "#hourly-duration", "The Pattern throughout A Day"))
)
),
# Location
tags$li(tags$a(href = "#location", "Location Analysis"),
tags$ul(
tags$li(tags$a(href = "#start-street", "Count of start street")),
tags$li(tags$a(href = "#end-street", "Count of end street")),
tags$li(tags$a(href = "#trip-pairs", "Count of trip pairs"))
)
)
)
),
mainPanel(
style = "padding-right: 100px; text-align: left;",

h1("Urban Pedals:", align = "center", id = "title"),
h3("Unveiling Toronto's Bike Share Story", align = "center"),
h2("Data Source", id = "source"),
div(
style = "font-size: 18px",
p("Historical Bike Share Toronto ridership data."),
p("Ref: ",tags$a(href ="https://open.toronto.ca/dataset/bike-share-toronto-ridership-data/", "Toronto Open Data")
),
),
tags$hr(),
div(style = "height: 50px"),

# data processing
h2("Data Processing", id = "processing"),
div(

```

```

        style = "font-size: 18px",
        p("Handle NA and 'NULL' Value"),
        p("Data Overview:"),
    ),
    br(),
    imageOutput("photo"),
    tags$hr(),

    # Trip Analysis
    h2("Trip analysis", id = "trip"),
    h3("Yearly Trends", id = "yearly-trip"),
    tableOutput('trip_yearly_tb'),
    br(),
    plotOutput("trip_yearly_plot"),
    br(),
    div(
        style = "font-size: 24px;",
        p("The annual trip count exhibits a consistent upward trend, doubling from approximately 2.4 million to 4.6 million over the specified period.")
    ),
    tags$hr(),

    # trip monlty
    h3("The Pattern over Months", id = "monthly-trip"),
    br(),
    DT::dataTableOutput("trip_monthly_tb"),
    br(),
    plotOutput("trip_monthly_plot"),
    div(
        style = "font-size: 24px;",
        # description
        p("")
    ),
    tags$hr(style = "padding-bottom: 50px"),

    # trip hourly
    h3("The Pattern throughout A Day", id = "hourly-trip"),
    br(),
    DT::dataTableOutput("trip_hourly_tb"),
    br(),
    plotOutput("trip_hourly_plot"),
    div(

```

```
      style = "font-size: 24px;",
      # description
      p("")
    ),
tags$hr(),

# Duration
h2("Duration Analysis", id = "duration"),

# duration yearly
h3("Yearly Trend", id = "yearly-duration"),
br(),
DT::dataTableOutput("duration_yearly_tb"),
br(),
plotOutput("duration_yearly_plot"),
div(
  style = "font-size: 24px;",
  # description
  p("")
),
tags$hr(),

# duration monthly
h3("The Pattern over Months", id = "monthly-duration"),
br(),
DT::dataTableOutput("duration_monthly_tb"),
br(),
plotOutput("duration_monthly_plot"),
div(
  style = "font-size: 24px;",
  # description
  p("")
),
tags$hr(),

# duration hourly
h3("The Pattern throughout A Day", id = "hourly-duration"),
br(),
DT::dataTableOutput("duration_hourly_tb"),
br(),
plotOutput("duration_hourly_plot"),
div(
```

```
      style = "font-size: 24px;",
      # description
      p("")
    ),
    tags$hr(),

    # Location
    h2("Location Analysis", id = "location"),

    # Location start street
    h3("Total Number of trips per starting stations", id = "start-street"),
    br(),
    DT::dataTableOutput("hot_start_streets_tb"),
    br(),
    plotOutput("hot_start_streets_plot"),
    div(
      style = "font-size: 24px;",
      # description
      p("")
    ),
    tags$hr(),

    # Location end street
    h3("Total Number of trips per ending street", id = "end-street"),
    br(),
    DT::dataTableOutput("hot_end_streets_tb"),
    br(),
    plotOutput("hot_end_streets_plot"),
    div(
      style = "font-size: 24px;",
      # description
      p("")
    ),
    tags$hr(),

    # Location pairs
    h3("Total Number of trips per origin-destination pair", id = "trip-pairs"),
    br(),
    DT::dataTableOutput("hot_trip_tb"),
    br(),
    plotOutput("hot_trip_plot"),
```

```

    div(
      style = "font-size: 24px;",
      # description
      p("")
    ),
    tags$hr(),

  )
)
)

server <- function(input, output) {

  output$photo <- renderImage({
    list(
      src = file.path("info.png"),
      contentType = "image/png",
      width = 800
    )
  }, deleteFile = FALSE)

  # trip yearly
  output$trip_yearly_tb <- renderTable(trip_yearly)
  output$trip_yearly_plot <- renderPlot({
    ggplot(
      data = trip_yearly,
      mapping = aes(
        x = Year,
        y = Trip,
        group = 1
      )
    ) +
    geom_line() +
    geom_point() +
    geom_text(aes(label = as.character(Trip)), vjust = 1, hjust = 0.5) +
    labs(
      title = "Yearly Trip Trend",
      x = "Year",
      y = "Trip"
    ) +
    theme_minimal()
  })
}

```

```

}))

# trip monthly
output$trip_monthly_tb <- DT::renderDataTable(DT::datatable({
  data <- trip_monthly_across_year
  data
})))

output$trip_monthly_plot <- renderPlot({
  ggplot(
    data = trip_monthly_across_year,
    mapping = aes(
      x = factor(Month),
      y = Trip,
      fill = factor(Month)
    )
  ) +
  geom_boxplot(show.legend = FALSE) +
  labs(
    title = "Monthly Distribution of Trips",
    x = "Month",
    y = "Trip",
    fill = "Month"
  ) +
  theme_minimal()
})

# trip hourly
output$trip_hourly_tb <- DT::renderDataTable(DT::datatable({
  data <- trip_hourly_across_month
  data
})))

output$trip_hourly_plot <- renderPlot({
  ggplot(
    trip_hourly_across_month,
    aes(
      x = factor(Hour),
      y = Trip,
      fill = factor(Hour)
    )
  ) +

```

```

    geom_boxplot(show.legend = FALSE) +
    labs(
    title = "Hourly Trip Distribution",
    x = "Hour of Day",
    y = "Number of Trips",
    fill = "Hour of Day") +
    theme_minimal()
  })

# duration yearly
output$duration_yearly_tb <- DT::renderDataTable(DT::datatable({
  data <- duration_yearly
  data
}))

output$duration_yearly_plot <- renderPlot({
  ggplot(
    data = duration_yearly,
    mapping = aes(
      x = Year,
      y = Mean_Duration,
      group = 1
    )
  ) +
  geom_line() +
  geom_point() +
  geom_text(
    aes(
      label = as.character(Mean_Duration)
    ),
    vjust = 1,
    hjust = 0.5
  ) +
  labs(
    title = "Average Duration Trend",
    x = "Year",
    y = "Average Duration"
  ) +
  theme_minimal()
})

```

```
# duration monthly
output$duration_monthly_tb <- DT::renderDataTable(DT::datatable({
  data <- duration_monthly_across_year
  data
}))
```

```
output$duration_monthly_plot <- renderPlot({
  ggplot(
    data = duration_monthly_across_year,
    mapping = aes(
      x = factor(Month),
      y = Mean_Duration,
      fill = factor(Month)
    )
  ) +
  geom_boxplot(show.legend = FALSE) +
  labs(
    title = "Monthly Duration Distribution",
    x = "Month",
    y = "Mean Duration (minutes)"
  ) +
  theme_minimal()
})
```

```
# duration hourly
output$duration_hourly_tb <- DT::renderDataTable(DT::datatable({
  data <- duration_hourly_across_month
  data
}))
```

```
output$duration_hourly_plot <- renderPlot({
  ggplot(
    data = duration_hourly_across_month,
    mapping = aes(
      x = factor(Hour),
      y = Mean_Duration,
      fill = factor(Hour)
    )
  ) +
  geom_boxplot(show.legend = FALSE) +
  labs(
```



```

        title = "Hourly Duration Distribution",
        x = "Hour of Day",
        y = "Mean Duration (minutes)"
    ) +
    theme_minimal()
  })

# Location start street
output$hot_start_streets_tb <- DT::renderDataTable(DT::datatable({
  data <- hot_start_streets
  data
}))

output$hot_start_streets_plot <- renderPlot({
  ggplot(
    data = hot_start_streets,
    mapping = aes(
      x = reorder(Start.Station.Street, -Total_Trips),
      y = Total_Trips,
      fill = reorder(Start.Station.Street, -Total_Trips)
    )
  ) +
  geom_bar(
    stat = "identity"
  ) +
  labs(
    title = "Total Number of trips per starting street",
    x = "Starting street",
    y = "Number of trips",
    fill = "Starting street"
  ) +
  theme_minimal()+
  theme(axis.text.x = element_blank())
})

# Location end street
output$hot_end_streets_tb <- DT::renderDataTable(DT::datatable({
  data <- hot_end_streets
  data
}))

output$hot_end_streets_plot <- renderPlot({

```

```

ggplot(
  data = hot_end_streets,
  mapping = aes(
    x = reorder(End.Station.Street, -Total_Trips),
    y = Total_Trips,
    fill = reorder(End.Station.Street, -Total_Trips)
  )
) +
geom_bar(
  stat = "identity"
) +
labs(
  title = "Total Number of trips per ending street",
  x = "Ending street",
  y = "Number of trips",
  fill = "Ending street"
) +
theme_minimal()+
theme(axis.text.x = element_blank()))
})

# Location pairs
output$hot_trip_tb <- DT::renderDataTable(DT::datatable({
  data <- hot_trip
  data
}))

output$hot_trip_plot <- renderPlot({
  ggplot(
    data = hot_trip,
    mapping = aes(
      x = reorder(Trip.Spots, -Total_Trips),
      y = Total_Trips,
      fill = reorder(Trip.Spots, -Total_Trips)
    )
  ) +
  geom_bar(
    stat = "identity"
  ) +
  labs(
    title = "Total Number of trips per origin-destination pair",
    x = "Origin-destination pair",

```

```

      y = "Number of trips",
      fill = "Origin-destination pair"
    ) +
    theme_minimal()+
    theme(axis.text.x = element_blank() ,
          legend.text = element_text(size = 6))
  })
}

shinyApp(ui, server)

```

## Urban Pedals

- Data Source
- Data Processing
- Trip Analysis
  - Yearly Trends
  - The Pattern over Months
  - The Pattern throughout A Day
- Duration Analysis
  - Yearly Trend
  - The Pattern over Months
  - The Pattern throughout A Day
- Location Analysis
  - Count of start street
  - Count of end street
  - Count of trip pairs

# Urban Pedals:

## Unveiling Toronto's Bike Share Story

### Data Source

Historical Bike Share Toronto ridership data.

Ref: Toronto Open Data

(<https://open.toronto.ca/dataset/bike-share-toronto-ridership-data/>)