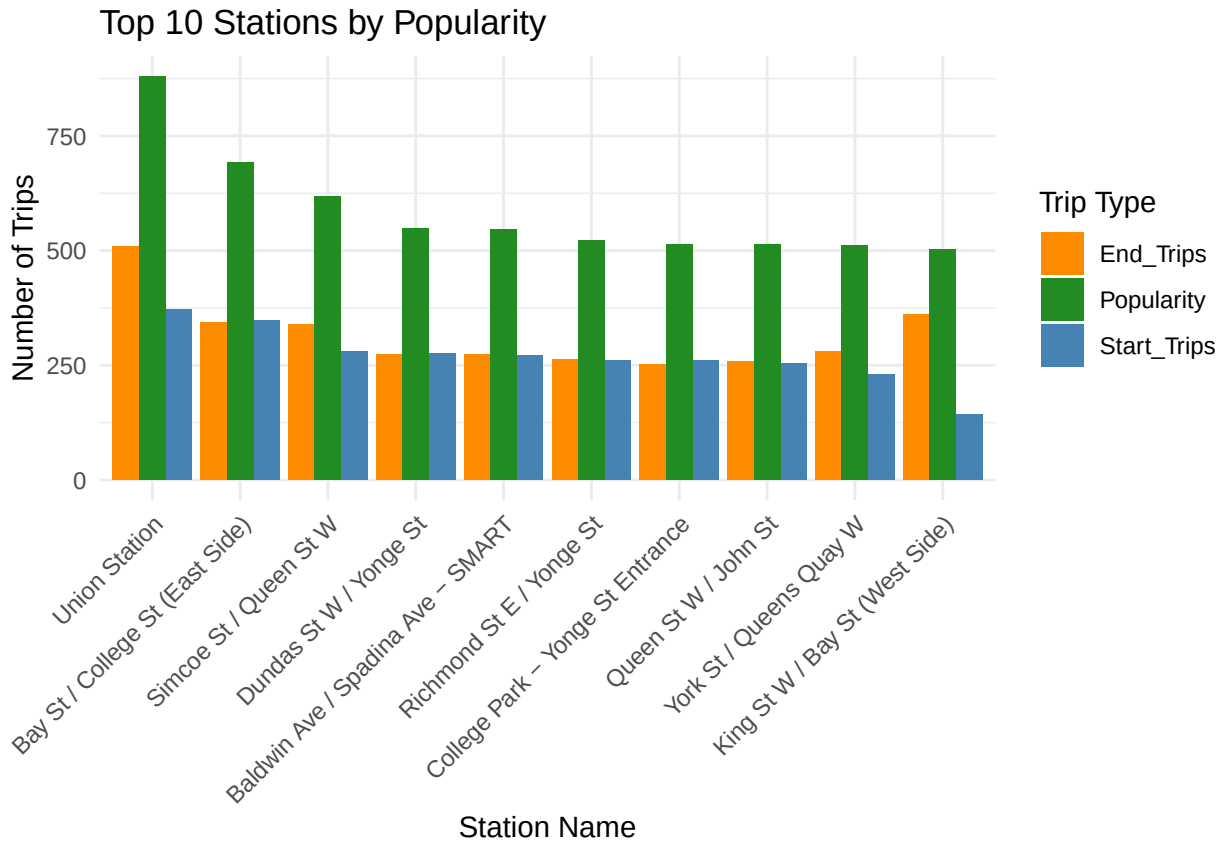
Bike Hackathon

2024-03-10

```
# Create an excel sheet of the most popular stations by start and end trips

# Calculate total trips started at each station
trips_started <- df %>%
  group_by(Start_Station_Id) %>%
  summarise(Start_Trips = n(), .groups = 'drop')

# Calculate total trips ended at each station
trips_ended <- df %>%
  group_by(End_Station_Id) %>%
  summarise(End_Trips = n(), .groups = 'drop')

# Combine the counts to a single dataframe
station_trips <- full_join(trips_started, trips_ended, by = c("Start_Station_Id" = "End_Station_Id")) %
  replace_na(list(Start_Trips = 0, End_Trips = 0)) %>%
  mutate(Total_Trips = Start_Trips + End_Trips) %>%
  arrange(desc(Total_Trips))

# Join with station names
popular_stations <- station_trips %>%
  left_join(station_name_mapping, by = c("Start_Station_Id" = "Station_Id"))

# Select and rename for final output
popular_stations <- popular_stations %>%
  select(Station_Id = Start_Station_Id, Station_Name, Start_Trips, End_Trips, Popularity = Total_Trips)

# Write the data to an Excel file
write.xlsx(popular_stations, "Most_Popular_Stations.xlsx")
```

```
#Plot start trips, end trips and popularity for top 10 stations

# Find the top 10 stations by total popularity
top_stations <- popular_stations %>%
  top_n(10, wt = Popularity)

# Reshape the data for ggplot
top_stations_long <- top_stations %>%
  gather(key = "Type", value = "Count", Start_Trips, End_Trips, Popularity)

# Create the triple bar chart
ggplot(top_stations_long, aes(x = reorder(Station_Name, -Count), y = Count, fill = Type)) +
  geom_bar(stat = "identity", position = "dodge") +
  scale_fill_manual(values = c("Start_Trips" = "steelblue", "End_Trips" = "darkorange", "Popularity" =
  labs(title = "Top 10 Stations by Popularity",
```

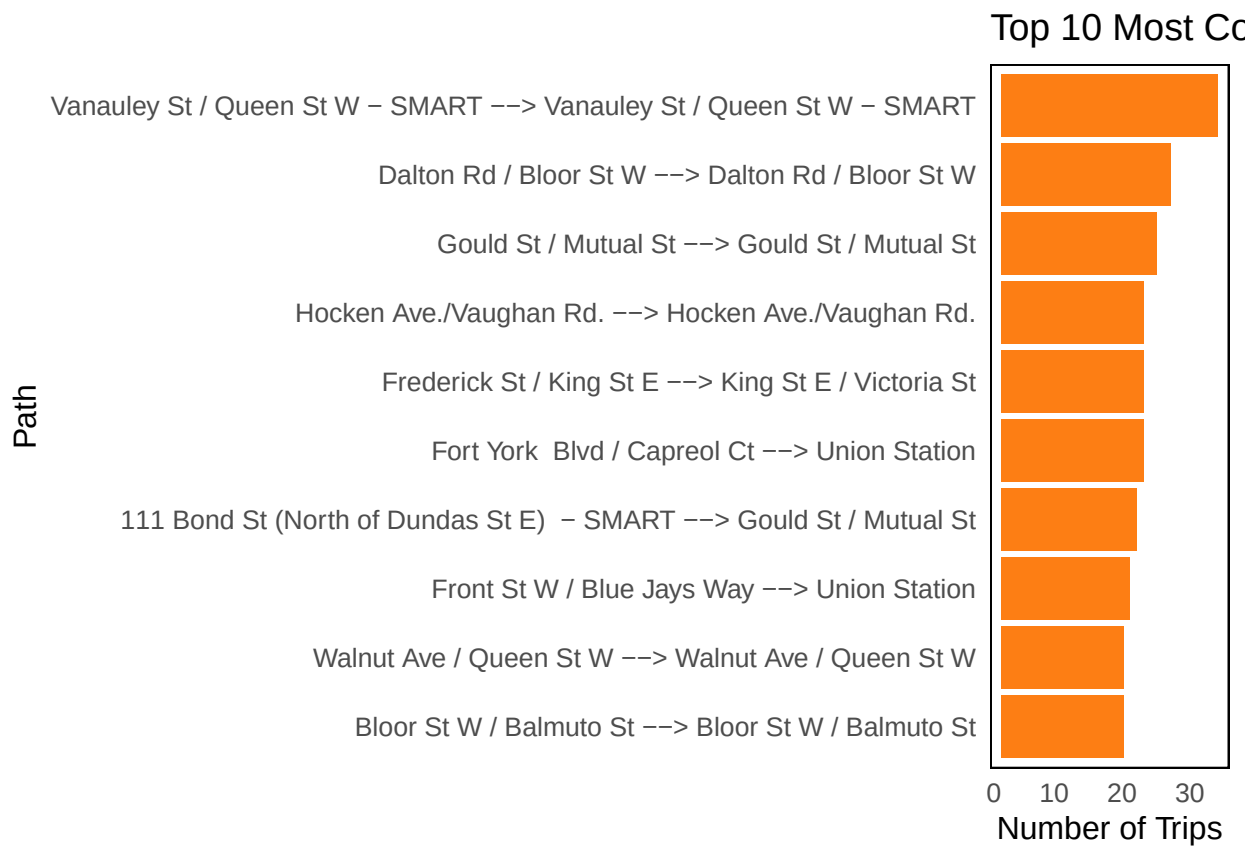
```
      x = "Station Name",
      y = "Number of Trips",
      fill = "Trip Type") +
  theme_minimal() +
  theme(
    panel.background = element_rect(fill = "white", colour = "white"), # Ensure white panel background
    plot.background = element_rect(fill = "white", colour = "white"), # Ensure white plot background
    legend.background = element_rect(fill = "white", colour = "white"), # Ensure white legend backgroun
    axis.text.x = element_text(angle = 45, hjust = 1)
  )
```




```
# Save the graph to a file
ggsave("Top_10_Stations_Popularity.png", width = 12, height = 8, units = "in", bg = "white")
```

```
# Visualize the top 10 most common paths with names
top_paths <- head(common_paths, 10)
# Adjusted plot code for a white background and improved styling
ggplot(top_paths, aes(x = reorder(paste(Start_Station_Name, End_Station_Name, sep = " --> "), Path_Coun
  geom_bar(stat = "identity", fill = "#FD7E14") +  # A pleasant shade of orange
  coord_flip() +  # Flip the axes for better readability
  labs(title = "Top 10 Most Common Paths", x = "Path", y = "Number of Trips") +
  theme_minimal(base_size = 12) +  # Use a minimal theme as the base
  theme(
    panel.background = element_rect(fill = "white", colour = "black"),  # White panel background
    plot.background = element_rect(fill = "white", colour = NA),  # White plot background
```

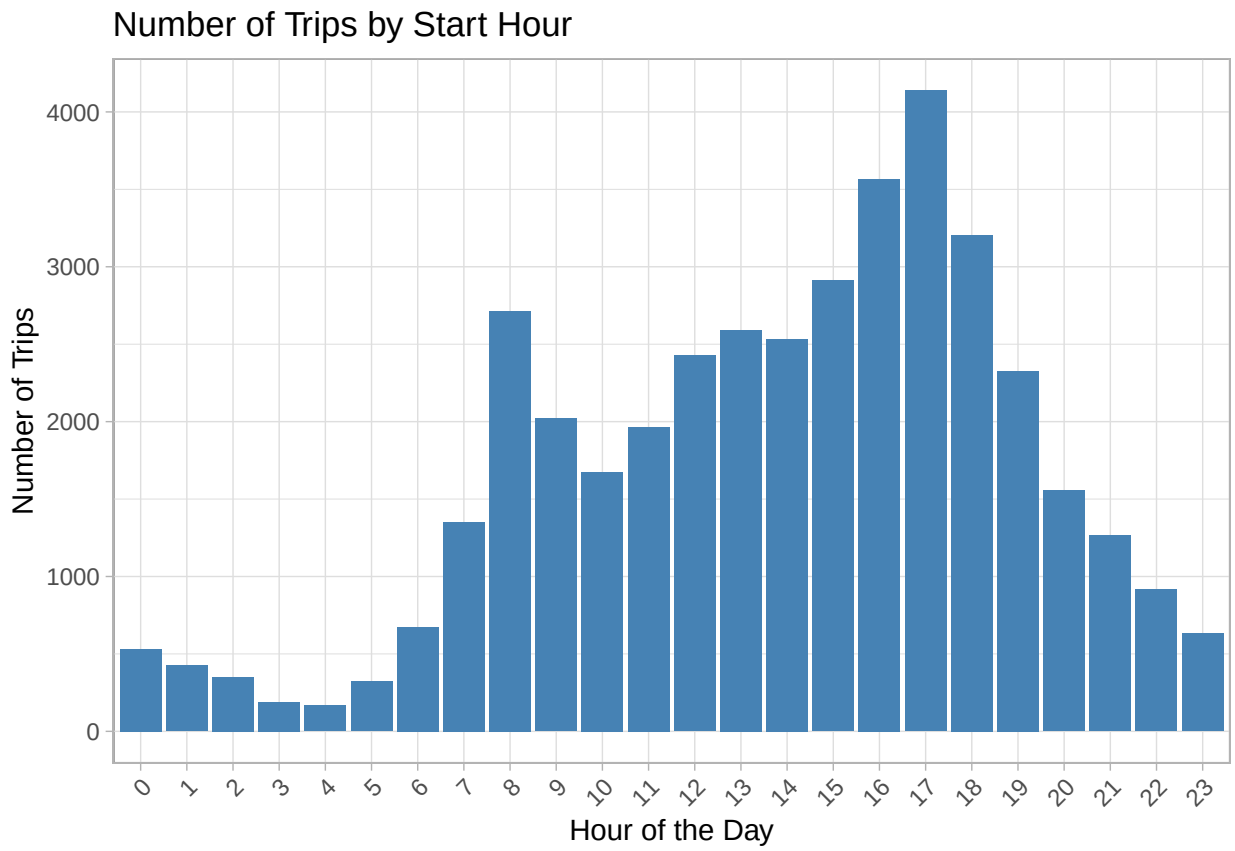
```
    panel.grid.major = element_blank(),  # Remove major grid lines
    panel.grid.minor = element_blank(),  # Remove minor grid lines
    axis.text.x = element_text(angle = 0, hjust = 1),  # Angle the x-axis text for readability
    legend.position = "none"  # Remove the legend if not necessary
  )
```




```
# Save the plot
ggsave("top_paths_styled.png", width = 10, height = 8, bg = "white")
```

```
# Visualize peak hours data

# Create peak hours data
peak_hours <- df %>%
  mutate(Hour = hour(Start_Time)) %>%
  group_by(Hour) %>%
  summarise(Trips = n(), .groups = 'drop') %>%
  arrange(Hour)

# Now create the plot
ggplot(peak_hours, aes(x = as.factor(Hour), y = Trips)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  theme_light() +  # Light theme for better contrast
  labs(title = "Number of Trips by Start Hour", x = "Hour of the Day", y = "Number of Trips") +
  theme(
    panel.background = element_rect(fill = "white", colour = "black"),
```

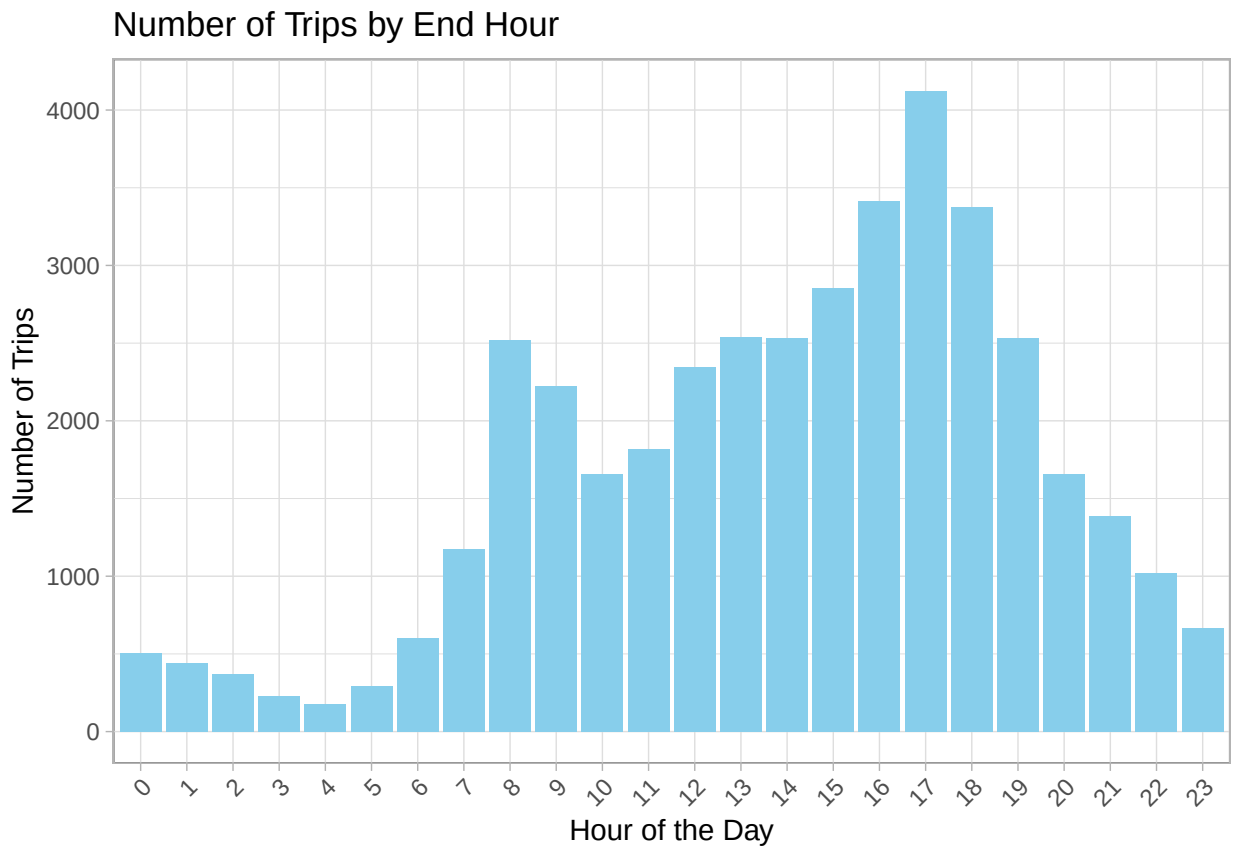
```
    plot.background = element_rect(fill = "white", colour = NA),
    axis.text.x = element_text(angle = 45, hjust = 1)  # Adjust angle of x labels if needed
  )
```




```
# Save the peak hours plot
ggsave("peak_start_hours.png", width = 10, height = 6, bg = "white")
```

```
# Visualize peak end hours data

# Create peak end hours data
peak_end_hours <- df %>%
  mutate(Hour = hour(End_Time)) %>%
  group_by(Hour) %>%
  summarise(Trips = n(), .groups = 'drop') %>%
  arrange(Hour)

# Now create the plot for end hours
ggplot(peak_end_hours, aes(x = as.factor(Hour), y = Trips)) +
  geom_bar(stat = "identity", fill = "skyblue") +  # A different fill color for distinction
  theme_light() +  # Light theme for better contrast
  labs(title = "Number of Trips by End Hour", x = "Hour of the Day", y = "Number of Trips") +
  theme(
    panel.background = element_rect(fill = "white", colour = "black"),
    plot.background = element_rect(fill = "white", colour = NA),
    axis.text.x = element_text(angle = 45, hjust = 1)  # Adjust angle of x labels if needed
```

```
)
```

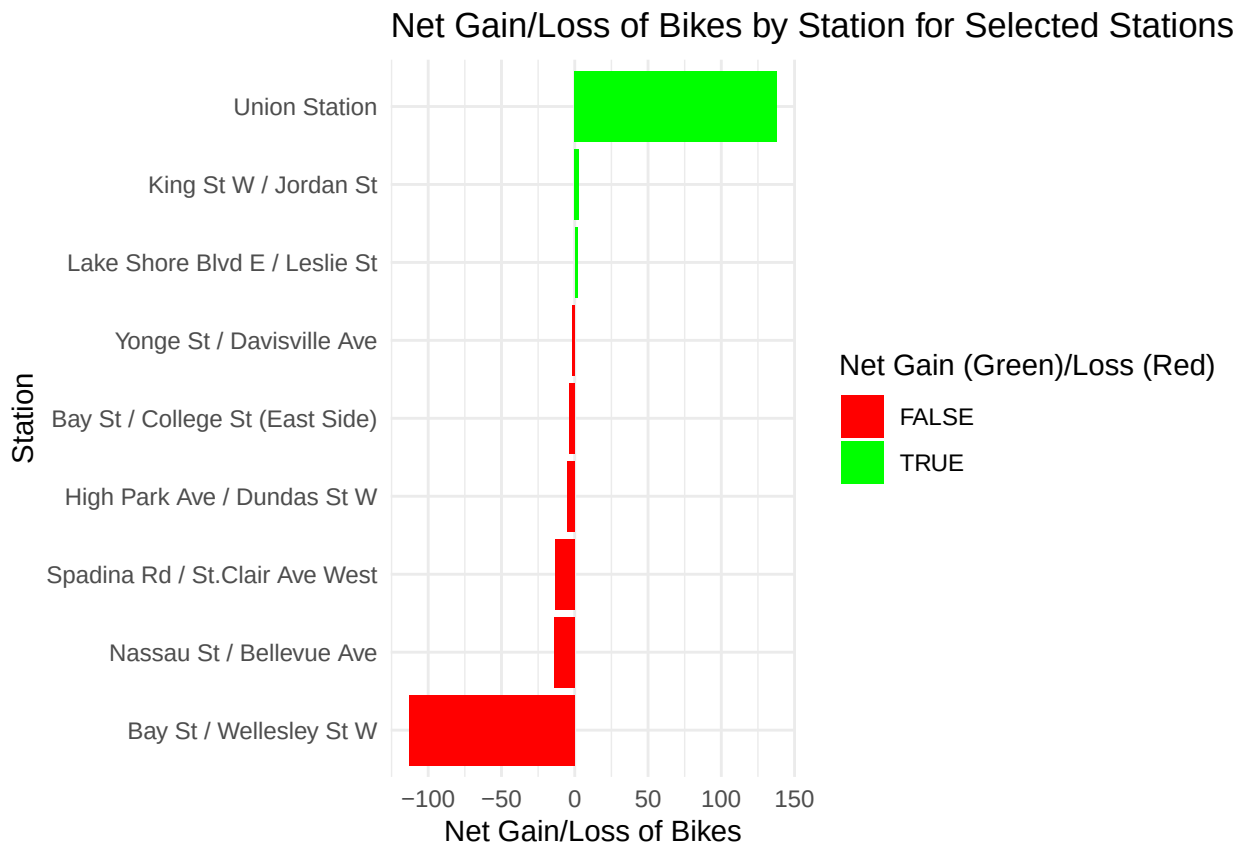



```
# Save the peak end hours plot
ggsave("peak_end_hours.png", width = 10, height = 6, bg = "white")
```

```
# This is to make a excel of net gain and loss at each station

# Calculate total trips started at each station
trips_started <- df %>%
  group_by(Start_Station_Id) %>%
  summarise(Trips_Started = n(), .groups = 'drop')

# Calculate total trips ended at each station
trips_ended <- df %>%
  group_by(End_Station_Id) %>%
  summarise(Trips_Ended = n(), .groups = 'drop')

# Join datasets to compare
station_trips <- merge(trips_started, trips_ended, by.x = "Start_Station_Id", by.y = "End_Station_Id", a

# Replace NA with 0 for stations that do not appear in either start or end
station_trips[is.na(station_trips)] <- 0

# Calculate net gain/loss of bikes at each station
station_trips$Net_Gain_Loss <- station_trips$Trips_Ended - station_trips$Trips_Started
```

```r
# Join station names for readability
station_trips <- merge(station_trips, station_name_mapping, by.x = "Start_Station_Id", by.y = "Station_I
names(station_trips)[names(station_trips) == "Station_Name"] <- "Station_Name"

# Create table
write.xlsx(station_trips, "Net_Gain_Loss_Bikes_by_Station.xlsx")
```

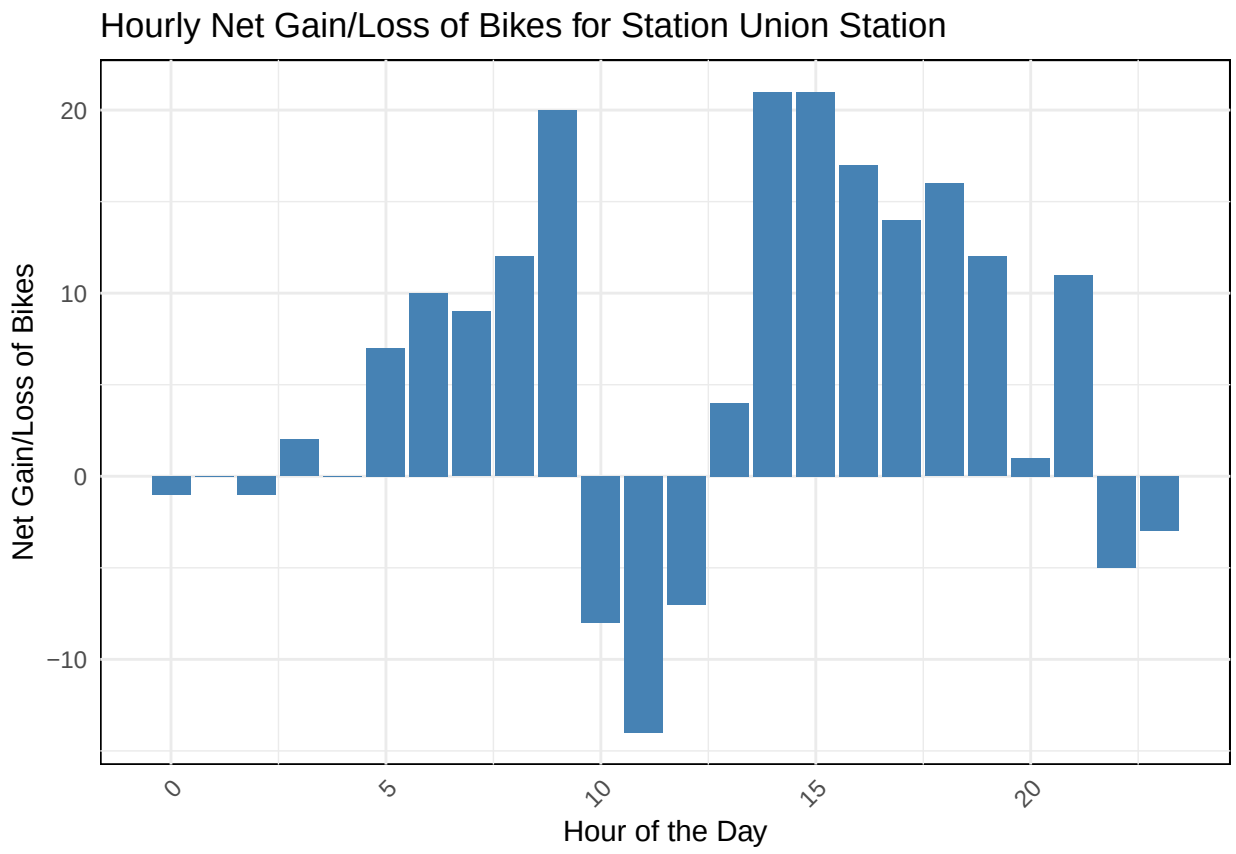
```r
# This chunk will show net gain and loss at the 9 stations shown in the case file to plot for easier re

# List of specific station IDs
specific_station_ids <- c(7033, 7417, 7581, 7640, 7357, 7006, 7378, 7543, 7030)

# Filter the dataframe for relevant stations
df_filtered <- df %>%
  filter(Start_Station_Id %in% specific_station_ids | End_Station_Id %in% specific_station_ids)

# Calculate total trips started at each of the specific stations
trips_started_specific <- df_filtered %>%
  filter(Start_Station_Id %in% specific_station_ids) %>%
  group_by(Start_Station_Id) %>%
  summarise(Trips_Started = n(), .groups = 'drop')

# Calculate total trips ended at each of the specific stations
trips_ended_specific <- df_filtered %>%
  filter(End_Station_Id %in% specific_station_ids) %>%
  group_by(End_Station_Id) %>%
  summarise(Trips_Ended = n(), .groups = 'drop')

# Join datasets to compare
station_trips_specific <- merge(trips_started_specific, trips_ended_specific, by.x = "Start_Station_Id"

# Replace NA with 0 for stations that do not appear in either start or end
station_trips_specific[is.na(station_trips_specific)] <- 0

# Calculate net gain/loss of bikes at each station
station_trips_specific$Net_Gain_Loss <- station_trips_specific$Trips_Ended - station_trips_specific$Trip

# Join station names for readability
station_trips_specific <- merge(station_trips_specific, station_name_mapping, by.x = "Start_Station_Id"
names(station_trips_specific)[names(station_trips_specific) == "Station_Name"] <- "Station_Name"

# For visualization, a simple bar chart for these specific stations
ggplot(station_trips_specific, aes(x = reorder(Station_Name, Net_Gain_Loss), y = Net_Gain_Loss, fill = N
  geom_bar(stat = "identity") +
  scale_fill_manual(values = c("TRUE" = "green", "FALSE" = "red"), name = "Net Gain (Green)/Loss (Red)")
  coord_flip() +
  labs(title = "Net Gain/Loss of Bikes by Station for Selected Stations", x = "Station", y = "Net Gain/L
  theme_minimal()
```



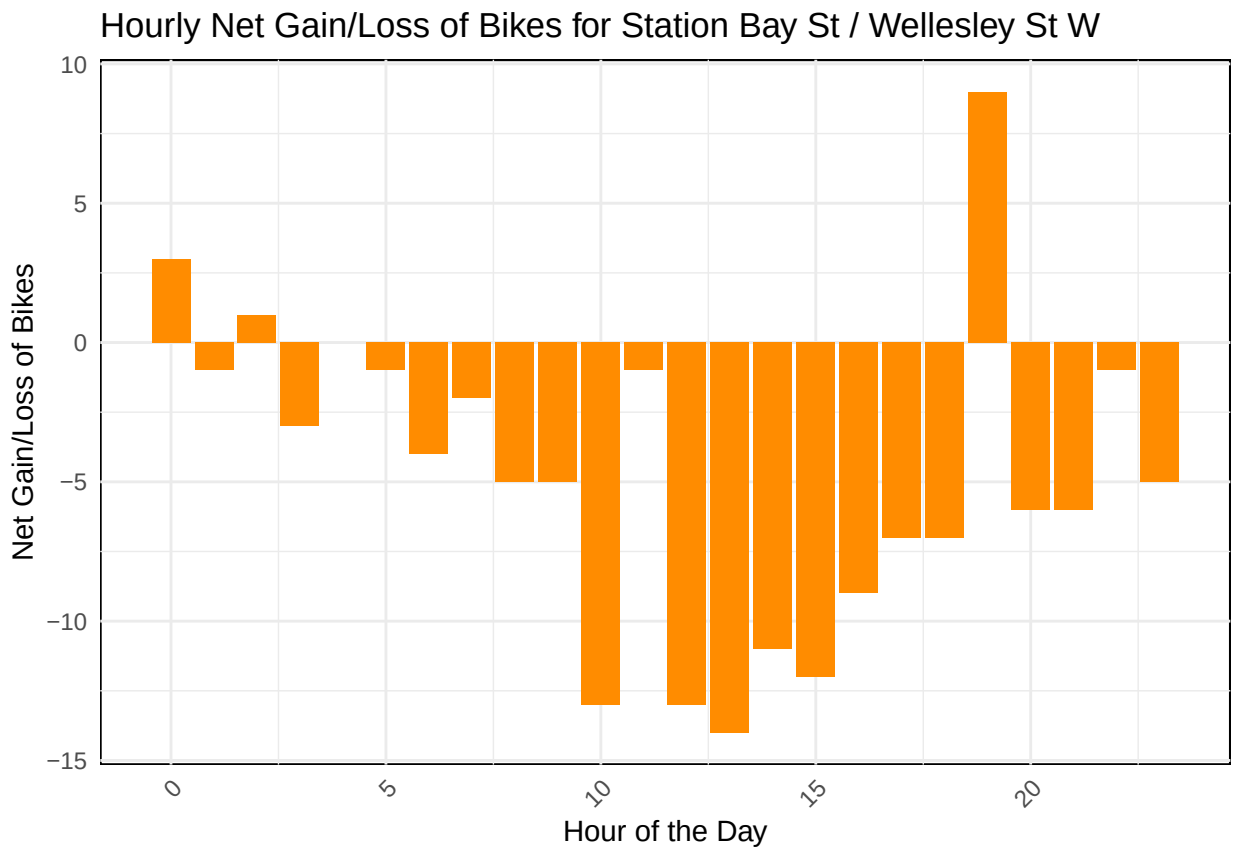

```
# Save the plot
ggsave("net_gain_loss_bikes_by_selected_stations.png", width = 12, height = 10, bg = "white")
```

```
#Plot net gain and loss by hour for the biggest loss station and gain station

# Specific station IDs for Union Station and Bay St/Wellesley
specific_station_ids <- c(7033, 7030)

# Filter the data for these stations
specific_stations_data <- df %>%
  filter(Start_Station_Id %in% specific_station_ids | End_Station_Id %in% specific_station_ids)

# Calculate trips started each hour for the specific stations
hourly_trips_started <- specific_stations_data %>%
  filter(Start_Station_Id %in% specific_station_ids) %>%
  mutate(Start_Hour = hour(Start_Time)) %>%
  group_by(Start_Hour, Start_Station_Id) %>%
  summarise(Trips_Started = n(), .groups = 'drop')

# Calculate trips ended each hour for the specific stations
hourly_trips_ended <- specific_stations_data %>%
  filter(End_Station_Id %in% specific_station_ids) %>%
  mutate(End_Hour = hour(End_Time)) %>%
  group_by(End_Hour, End_Station_Id) %>%
  summarise(Trips_Ended = n(), .groups = 'drop')
```

```r
# Merge the start and end trip data frames
hourly_net <- merge(hourly_trips_started, hourly_trips_ended,
                    by.x = c("Start_Hour", "Start_Station_Id"),
                    by.y = c("End_Hour", "End_Station_Id"),
                    all = TRUE)

# Replace NA with 0 for hours that do not have any trips started or ended
hourly_net[is.na(hourly_net)] <- 0

# Calculate the net gain/loss for each station and hour
hourly_net$Net_Gain_Loss <- hourly_net$Trips_Ended - hourly_net$Trips_Started

# Join station names for readability
hourly_net <- merge(hourly_net, station_name_mapping, by.x = "Start_Station_Id", by.y = "Station_Id", al

# Rename columns for clarity
names(hourly_net)[names(hourly_net) == "Station_Name"] <- "Station_Name"
hourly_net$Hour <- ifelse(!is.na(hourly_net$Start_Hour), hourly_net$Start_Hour, hourly_net$End_Hour)

hourly_net_long <- hourly_net %>%
  select(Station_Id = Start_Station_Id, Hour, Net_Gain_Loss, Station_Name) %>%
  gather(key = "Type", value = "Value", c("Net_Gain_Loss"))

# First, we prepare the data for Station 7033
hourly_net_7033 <- specific_stations_data %>%
  filter(Start_Station_Id == 7033 | End_Station_Id == 7033) %>%
  mutate(Hour = hour(Start_Time)) %>%
  group_by(Hour) %>%
  summarise(
    Trips_Started = sum(Start_Station_Id == 7033),
    Trips_Ended = sum(End_Station_Id == 7033),
    .groups = 'drop'
  ) %>%
  mutate(Net_Gain_Loss = Trips_Ended - Trips_Started)

# Then plot for Station 7033
ggplot(hourly_net_7033, aes(x = Hour, y = Net_Gain_Loss)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(title = "Hourly Net Gain/Loss of Bikes for Station Union Station",
       x = "Hour of the Day", y = "Net Gain/Loss of Bikes") +
  theme_minimal() +
  theme(
    panel.background = element_rect(fill = "white", colour = "black"),
    plot.background = element_rect(fill = "white", colour = "white"),
    axis.text.x = element_text(angle = 45, hjust = 1)
  )
```




```r
# Save the plot for Station 7033
ggsave("hourly_net_gain_loss_7033.png", width = 10, height = 6, units = "in", bg = "white")

# Now for Station 7030
hourly_net_7030 <- specific_stations_data %>%
  filter(Start_Station_Id == 7030 | End_Station_Id == 7030) %>%
  mutate(Hour = hour(Start_Time)) %>%
  group_by(Hour) %>%
  summarise(
    Trips_Started = sum(Start_Station_Id == 7030),
    Trips_Ended = sum(End_Station_Id == 7030),
    .groups = 'drop'
  ) %>%
  mutate(Net_Gain_Loss = Trips_Ended - Trips_Started)

# Then plot for Station 7030
ggplot(hourly_net_7030, aes(x = Hour, y = Net_Gain_Loss)) +
  geom_bar(stat = "identity", fill = "darkorange") +
  labs(title = "Hourly Net Gain/Loss of Bikes for Station Bay St / Wellesley St W",
       x = "Hour of the Day", y = "Net Gain/Loss of Bikes") +
  theme_minimal() +
  theme(
    panel.background = element_rect(fill = "white", colour = "black"),
    plot.background = element_rect(fill = "white", colour = "white"),
    axis.text.x = element_text(angle = 45, hjust = 1)
  )
```




```
# Save the plot for Station 7030
ggsave("hourly_net_gain_loss_7030.png", width = 10, height = 6, units = "in", bg = "white")
```