Appendix I

1. Load Libraries

```
knitr::opts_chunk$set(echo = TRUE, message = FALSE, warning = FALSE)
library(tidyverse)
library(lubridate)
library(forecast)
library(tseries)
library(vars)
library(gridExtra)
library(kableExtra)
library(tsibble)
library(fable)
library(DT)
```

# 2. Load and Prepare Data

```
library(readr)
raw <- read_csv("products_02.csv",
    col_types = cols(stock_xfer_date = col_date(format = "%m/%d/%Y")))

head(raw,5)
```

```
# -------------------------------
# 3 Clean and split dimensions (remove " cm")
# -------------------------------

raw <- raw %>%
  mutate(
    stock_xfer_date = ymd(stock_xfer_date),
    prod_dimensions = str_remove(prod_dimensions, " cm")
  ) %>%
  separate(prod_dimensions, into = c("len_cm", "width_cm", "height_cm"), sep = "x", convert =

# Compute volume per item in cm3, convert to m3, compute space used
raw <- raw %>%
  mutate(
    volume_cm3 = len_cm * width_cm * height_cm,
    space_used_cm3 = volume_cm3 * stock_qty,
    space_used_m3 = space_used_cm3 / 1e6
  )
```

```
# -------------------------------
# 3. Create Daily Aggregated Time Series
# -------------------------------

daily_totals <- raw %>%
  group_by(stock_xfer_date) %>%
  summarise(
    total_stock_qty = sum(stock_qty),
    total_space_m3 = sum(space_used_m3)
  ) %>%
  arrange(stock_xfer_date)

# Create ts object (daily frequency assumed to be 365)
ts_total_space <- ts(daily_totals$total_space_m3, frequency = 7)   # weekly seasonality
ts_total_stock <- ts(daily_totals$total_stock_qty, frequency = 7)


ggplot(daily_totals, aes(stock_xfer_date, total_space_m3)) +
geom_line(color = "steelblue") +
labs(title="Total Space Used (m³) Over Time",
x="Date", y="Space Used (m³)") +
theme_minimal()
```

```
# -------------------------------
# 4. Create Category-Level Time Series (FIXED)
# -------------------------------

cat_daily <- raw %>%
  group_by(stock_xfer_date, prod_cat) %>%
  summarise(space_m3 = sum(space_used_m3), stock_qty = sum(stock_qty)) %>%
  ungroup()

cat_space_wide <- cat_daily %>%
  dplyr::select(stock_xfer_date, prod_cat, space_m3) %>%
  tidyr::pivot_wider(names_from = prod_cat, values_from = space_m3) %>%
  dplyr::arrange(stock_xfer_date)


# Replace NA (missing categories on some days)
```

```
cat_space_wide[is.na(cat_space_wide)] <- 0

cat("Category-wide dataset created:\n")
print(head(cat_space_wide))
print(dim(cat_space_wide))

# Convert to time series matrix
ts_cat_space <- ts(cat_space_wide[, -1], frequency = 7)

cat("ts_cat_space created successfully:\n")
print(dim(ts_cat_space))
print(colnames(ts_cat_space))



ggplot(cat_daily, aes(stock_xfer_date, space_m3, color=prod_cat)) +
geom_line() +
theme_minimal() +
labs(title="Daily Space Used by Category (m³)", x="Date", y="Space (m³)")
```

```
# -------------------------------
# 5. Stationarity Tests (ADF)
# -------------------------------

adf_total <- adf.test(ts_total_space)
adf_total

# First difference
adf_total_diff <- adf.test(diff(ts_total_space))
adf_total_diff

# Category ADF tests
apply(ts_cat_space, 2, adf.test)
```

```
adf_total_diff <- adf.test(diff(ts_total_space))
```

```
# -------------------------------
# 6. Fit SARIMA Model for Total Space
# -------------------------------

sarima_model <- Arima(ts_total_space, order = c(0,1,2),
                      seasonal = list(order = c(2,0,2), period = 7))

summary(sarima_model)



p1 <- ggAcf(residuals(sarima_model)) + ggtitle("ACF of SARIMA Residuals")
p2 <- ggPacf(residuals(sarima_model)) + ggtitle("PACF of SARIMA Residuals")
grid.arrange(p1, p2, ncol=2)

# Forecast next 60 days
sarima_fc <- forecast(sarima_model, h=60)

autoplot(sarima_fc) +
ggtitle("SARIMA Forecast - Total Space Used (m³)") +
theme_minimal()
```

```
# -------------------------------
# 7. Train/Test Accuracy Evaluation
# -------------------------------

n <- length(ts_total_space)

# Ensure at least 60 points exist
h <- min(60, floor(n * 0.2))   # 20% test set if data is shorter

cat("Series length:", n, "\n")
cat("Forecast horizon:", h, "\n")

# Safe slicing using vector positions
train <- ts_total_space[1:(n - h)]
test  <- ts_total_space[(n - h + 1):n]

sarima_train_model <- Arima(
  train,
  order = c(0,1,2),
  seasonal = list(order = c(2,0,2), period = 7)
)

pred <- forecast(sarima_train_model, h = h)
```

```r
cat("SARIMA Accuracy (Safe Train/Test Split):\n")
print(accuracy(pred, test))
```

```r
#Box.test(sarima_resid, lag = 10, type = "Ljung-Box")
```

```r
# -------------------------------
# 8. Fit VAR Model for Category-Level Space Usage
# -------------------------------
# First difference for stationarity
ts_cat_space_diff <- diff(ts_cat_space)

# Select optimal lag (VAR order)
lag_selection <- VARselect(ts_cat_space_diff, lag.max = 15, type = "const")
lag_selection

# Fit VAR(2) as recommended
var_model <- VAR(ts_cat_space_diff, p = 2, type = "const")
var_model
```

```r
# -------------------------------
# Correction for VAR Forecasting
# -------------------------------

cat_matrix <- data.frame(cat_space_wide[, -1])

for(i in seq_along(cat_matrix)){
  cat_matrix[[i]] <- as.numeric(cat_matrix[[i]])
}

cat_matrix_diff <- diff(as.matrix(cat_matrix))
colnames(cat_matrix_diff) <- colnames(cat_matrix)
cat_df_diff <- data.frame(cat_matrix_diff)
```

```r
library(vars)
var_model <- vars::VAR(cat_df_diff, p = 2, type = "const")
class(var_model)
```

```r
var_raw <- stats::predict(var_model, n.ahead = 30)
```

```r
fcst_means <- lapply(var_raw$fcst, function(x) as.numeric(x[,1]))

var_fc_diff <- do.call(cbind, fcst_means)
colnames(var_fc_diff) <- names(var_raw$fcst)
```

```r
last_levels <- as.numeric(tail(cat_matrix, 1))

var_fc_levels <- sweep(apply(var_fc_diff, 2, cumsum), 2, last_levels, "+")
var_fc_levels <- as.data.frame(var_fc_levels)
```

```r
# -------------------------------
# 9. VAR Forecasting (Category Space)
# -------------------------------

var_forecast <- predict(var_model, n.ahead = 30)

# Rebuild into levels
initial_values <- tail(cat_space_wide[, -1], 1)

var_fc_levels <- sapply(var_forecast$fcst, function(x) {
  cumsum(x[, 1]) + as.numeric(initial_values)
})

var_fc_levels <- as.data.frame(var_fc_levels)

cat("Category-level VAR forecast (levels):\n")
print(var_fc_levels[1:5, ])
```

```r
var_fc <- predict(var_model, n.ahead = 30)

initial_values <- tail(cat_space_wide[, -1], 1)

var_fc_levels <- sapply(var_fc$fcst, function(x) {
cumsum(x[,1]) + as.numeric(initial_values)
})

var_fc_levels <- as.data.frame(var_fc_levels)
head(var_fc_levels)
```

```r
matplot(var_fc_levels, type="l", lwd=2, lty=1,
col=c("purple","blue","green","red"),
ylab="Space (m³)", xlab="Day Ahead",
main="Category-Level VAR Forecast (Space Used)")
legend("topleft", legend=colnames(var_fc_levels),
col=c("purple","blue","green","red"), lty=1, lwd=2)
```

```r
# -------------------------------
# 10. Hierarchical Reconciliation
# -------------------------------
# Scale category forecasts to match SARIMA total forecast
cat_matrix <- cat_space_wide[, -1]               # remove date
cat_matrix <- data.frame(cat_matrix)             # remove tibble class

# Force everything numeric
for(i in seq_along(cat_matrix)){
  cat_matrix[[i]] <- as.numeric(cat_matrix[[i]])
}

# Check structure
str(cat_matrix)

# Differencing
cat_matrix_diff <- diff(as.matrix(cat_matrix))  # returns pure numeric matrix
colnames(cat_matrix_diff) <- colnames(cat_matrix)

# Convert to dataframe for VAR
cat_df_diff <- as.data.frame(cat_matrix_diff)

# Final check before VAR
cat("CLEAN cat_df_diff:\n")
str(cat_df_diff)
```

```r
# Number of forecast steps in VAR output
h_var <- nrow(var_fc_levels)

# Take the first h_var SARIMA forecasts and coerce to numeric
sarima_vals <- as.numeric(sarima_fc$mean[seq_len(h_var)])

# Sum of category forecasts per horizon
```

```
cat_sums <- rowSums(var_fc_levels)

# Scaling factor so that row sums match SARIMA totals
scale_factor <- sarima_vals / cat_sums

# Apply scaling to each row (each horizon)
reconciled_cat <- sweep(var_fc_levels, 1, scale_factor, "*")

# Inspect first few reconciled rows
head(reconciled_cat) %>% kable() %>% kable_styling()
```

```
sarima_vals <- as.numeric(sarima_fc$mean[seq_len(h_var)])
```

```
############################################################
# SUMMARY STATISTICS FOR INVENTORY & SPACE DATA
############################################################

library(dplyr)
library(tidyr)

# ---------------------------------------------------------
# 1. Summary of the raw dataset
# ---------------------------------------------------------
cat("==== RAW DATA SUMMARY ====\n")
print(summary(raw))

cat("\n==== RAW DATA STRUCTURE ====\n")
str(raw)
```

```
# ---------------------------------------------------------
# 2. Summary of per-unit volume and space
# ---------------------------------------------------------
#cat("\n==== VOLUME & SPACE SUMMARY (PER SKU TRANSFER) ====\n")
#raw %>%
#  select(volume_cm3, space_used_cm3, space_used_m3, stock_qty) %>%
# summary() %>%
#  print()
```

```
# ---------------------------------------------------------
# 3. Summary statistics for daily totals
# ---------------------------------------------------------
```

```r
daily_summary <- daily_totals %>%
  summarise(
    mean_stock = mean(total_stock_qty),
    sd_stock = sd(total_stock_qty),
    min_stock = min(total_stock_qty),
    max_stock = max(total_stock_qty),
    median_stock = median(total_stock_qty),
    mean_space = mean(total_space_m3),
    sd_space = sd(total_space_m3),
    min_space = min(total_space_m3),
    max_space = max(total_space_m3),
    median_space = median(total_space_m3)
  )

cat("\n==== DAILY TOTAL SUMMARY ====\n")
print(daily_summary)
```

```r
# ---------------------------------------------------------
# 4. Summary by product category (stock + space)
# ---------------------------------------------------------
cat("\n==== CATEGORY-LEVEL SUMMARY ====\n")

category_summary <- raw %>%
  group_by(prod_cat) %>%
  summarise(
    total_stock = sum(stock_qty),
    avg_stock = mean(stock_qty),
    sd_stock = sd(stock_qty),
    total_space_m3 = sum(space_used_m3),
    avg_space_m3 = mean(space_used_m3),
    sd_space_m3 = sd(space_used_m3),
    avg_volume_cm3 = mean(volume_cm3),
    sd_volume_cm3 = sd(volume_cm3)
  )

print(category_summary)
```

```r
# ---------------------------------------------------------
# 5. Summary of daily category-level space
# ---------------------------------------------------------
cat("\n==== DAILY CATEGORY SPACE SUMMARY ====\n")
```

```r
daily_cat_summary <- cat_space_wide %>%
  summarise(
    across(-stock_xfer_date,
           list(
             mean = mean,
             sd = sd,
             min = min,
             max = max,
             median = median
           ),
           .names = "{.col}_{.fn}"
    )
  )

print(daily_cat_summary)
```

```r
# ---------------------------------------------------------
# 6. Correlation analysis (optional)
# ---------------------------------------------------------
cat("\n==== CORRELATIONS ====\n")

cor_total <- cor(daily_totals$total_stock_qty, daily_totals$total_space_m3)
cat("Correlation (Total Stock vs Total Space):", cor_total, "\n")

cor_categories <- cor(cat_space_wide[, -1])
cat("\nCorrelation Matrix (Category Space m³):\n")
print(round(cor_categories, 3))
```

```r
############################################################
# QQ PLOTS FOR INVENTORY & SPACE TIME SERIES ANALYSIS
############################################################

library(ggplot2)

# ---------------------------------------------------------
# 1. QQ Plot for SARIMA residuals (Total Space)
# ---------------------------------------------------------
sarima_resid <- residuals(sarima_model)

# Base R QQ plot
qqnorm(sarima_resid, main = "QQ Plot of SARIMA Residuals (Total Space)")
```

```r
qqline(sarima_resid, col = "red", lwd = 2)

# ggplot version (optional)
sarima_df <- data.frame(resid = sarima_resid)

ggplot(sarima_df, aes(sample = resid)) +
  stat_qq() +
  stat_qq_line(col = "red", size = 1) +
  ggtitle("QQ Plot - SARIMA Residuals (Total Space)") +
  theme_minimal()
```

```r
# --------------------------------------------------------
# 2. QQ Plot for Total Space Series (Raw or Differenced)
# --------------------------------------------------------
# Raw series
qqnorm(ts_total_space, main = "QQ Plot - Total Space (Raw Data)")
qqline(ts_total_space, col = "red", lwd = 2)

# Differenced series
qqnorm(diff(ts_total_space), main = "QQ Plot - Differenced Total Space")
qqline(diff(ts_total_space), col = "blue", lwd = 2)
```

```r
# --------------------------------------------------------
# 3. QQ Plots for Category-Level Space (Each Category)
# --------------------------------------------------------
cat_names <- colnames(ts_cat_space)

par(mfrow = c(2, 2))  # 4 plots together

for (i in 1:ncol(ts_cat_space)) {
  qqnorm(ts_cat_space[, i],
         main = paste("QQ Plot -", cat_names[i]),
         cex = 0.8)
  qqline(ts_cat_space[, i], col = "red", lwd = 2)
}

par(mfrow = c(1, 1))  # reset layout
```

```r
# --------------------------------------------------------
# 4. QQ Plots for VAR Residuals (Category Model)
# --------------------------------------------------------
var_resid <- resid(var_model)
```

```r
cat("VAR residuals structure:\n")
print(str(var_resid))

# var_resid is a list with one matrix per equation - we loop through them
par(mfrow = c(2, 2))

# for (i in 1:length(var_resid)) {
#   v <- var_resid[[i]][, 1]  # extract residuals for equation i
#
#   qqnorm(v,
#          main = paste("QQ Plot - VAR Residuals:", names(var_resid)[i]),
#          cex = 0.8)
#   qqline(v, col = "red", lwd = 2)
# }
#
# par(mfrow = c(1, 1))
```

```r
############################################################
# TOTAL SPACE USAGE RESULTS (m³)
############################################################

library(dplyr)
library(ggplot2)
library(lubridate)
library(forecast)
```

```r
# ---------------------------------------------------------
# 1. Compute Daily Total Space Usage
# ---------------------------------------------------------

daily_totals <- raw %>%
  group_by(stock_xfer_date) %>%
  summarise(
    total_space_m3 = sum(space_used_m3),
    total_stock_qty = sum(stock_qty),
    .groups = "drop"
  ) %>%
  arrange(stock_xfer_date)

head(daily_totals)
```

```r
# ---------------------------------------------------------
# 2. Summary Statistics for Total Space
# ---------------------------------------------------------

total_space_summary <- daily_totals %>%
  summarise(
    mean_space = mean(total_space_m3),
    sd_space = sd(total_space_m3),
    min_space = min(total_space_m3),
    max_space = max(total_space_m3),
    median_space = median(total_space_m3),
    q1_space = quantile(total_space_m3, 0.25),
    q3_space = quantile(total_space_m3, 0.75),
    iqr_space = IQR(total_space_m3)
  )

cat("==== Total Space Usage Summary (m³/day) ====\n")
print(total_space_summary)
```

```r
# ---------------------------------------------------------
# 3. Summary Statistics for Total Stock Transfers
# ---------------------------------------------------------

total_stock_summary <- daily_totals %>%
  summarise(
    mean_stock = mean(total_stock_qty),
    sd_stock = sd(total_stock_qty),
    min_stock = min(total_stock_qty),
    max_stock = max(total_stock_qty),
    median_stock = median(total_stock_qty),
    q1_stock = quantile(total_stock_qty, 0.25),
    q3_stock = quantile(total_stock_qty, 0.75),
    iqr_stock = IQR(total_stock_qty)
  )

cat("\n==== Total Stock Transfer Summary (units/day) ====\n")
print(total_stock_summary)
```

```r
# ---------------------------------------------------------
# 4. Correlation Between Stock & Space
# ---------------------------------------------------------
```

```
cor_stock_space <- cor(daily_totals$total_stock_qty,
                       daily_totals$total_space_m3)

cat("\nCorrelation between Stock Transfers and Space Used:",
    round(cor_stock_space, 4), "\n")
```

```
# ---------------------------------------------------------
# 5. Plot: Total Daily Space Usage (m³)
# ---------------------------------------------------------

ggplot(daily_totals, aes(x = stock_xfer_date, y = total_space_m3)) +
  geom_line(color = "steelblue", linewidth = 1) +
  labs(
    title = "Daily Total Space Usage (m³) - 2023",
    x = "Date",
    y = "Space Used (m³)"
  ) +
  theme_minimal()
```

```
# ---------------------------------------------------------
# 6. Histogram + Density of Total Space Usage
# ---------------------------------------------------------

ggplot(daily_totals, aes(x = total_space_m3)) +
  geom_histogram(binwidth = 0.3, fill = "skyblue", color = "black") +
  geom_density(color = "red", linewidth = 1) +
  labs(
    title = "Distribution of Daily Space Usage",
    x = "Space Used (m³)",
    y = "Frequency"
  ) +
  theme_minimal()
```

```
# ---------------------------------------------------------
# 7. Prepare Time-Series Object for Modeling
# ---------------------------------------------------------

ts_total_space <- ts(daily_totals$total_space_m3, frequency = 7)

cat("\n==== Time Series Object for Total Space ====\n")
print(ts_total_space)
```

```
############################################################
# STOCK VS. SPACE CORRELATION
############################################################

library(dplyr)
library(ggplot2)
```

```
# ----------------------------------------------------------
# 1. Calculate correlation coefficient
# ----------------------------------------------------------

cor_stock_space <- cor(daily_totals$total_stock_qty,
                       daily_totals$total_space_m3)

cat("Correlation between Stock Transfers and Space Used (m³):",
    round(cor_stock_space, 4), "\n")
```

```
# ----------------------------------------------------------
# 2. Perform a correlation test (p-value, CI)
# ----------------------------------------------------------

cor_test_result <- cor.test(daily_totals$total_stock_qty,
                            daily_totals$total_space_m3)

cat("\n==== Correlation Test Results ====\n")
print(cor_test_result)
```

```
# ----------------------------------------------------------
# 3. Scatterplot of Stock vs Space
# ----------------------------------------------------------

ggplot(daily_totals, aes(x = total_stock_qty, y = total_space_m3)) +
  geom_point(color = "steelblue", alpha = 0.6) +
  geom_smooth(method = "lm", se = TRUE, color = "red", linewidth = 1) +
  labs(
    title = "Relationship Between Stock Transfers and Space Used",
    subtitle = paste("Correlation =", round(cor_stock_space, 3)),
    x = "Total Stock Transfers (Units)",
    y = "Total Space Used (m³)"
  ) +
  theme_minimal()
```

```
# -----------------------------------------------------------
# 4. Optional: Display scatterplot with log scaling
#    (useful if distribution is wide)
# -----------------------------------------------------------

ggplot(daily_totals, aes(x = total_stock_qty, y = total_space_m3)) +
  geom_point(color = "darkgreen", alpha = 0.6) +
  geom_smooth(method = "lm", color = "red") +
  scale_x_continuous(trans = "log10") +
  labs(
    title = "Stock vs Space (Log-Scaled Stock)",
    x = "Total Stock Transfers (log10)",
    y = "Total Space Used (m³)"
  ) +
  theme_minimal()
```

```
#############################################################
# CATEGORY-LEVEL RESULTS (SPACE + STOCK)
#############################################################

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

```
# -----------------------------------------------------------
# 1. Create daily category-level dataset
# -----------------------------------------------------------

cat_daily <- raw %>%
  group_by(stock_xfer_date, prod_cat) %>%
  summarise(
    space_m3 = sum(space_used_m3),
    stock_qty = sum(stock_qty),
    .groups = "drop"
  ) %>%
  arrange(stock_xfer_date)

head(cat_daily)
```

```
# ---------------------------------------------------------
# 2. Summary statistics per category (space + stock)
# ---------------------------------------------------------

cat_summary <- cat_daily %>%
  group_by(prod_cat) %>%
  summarise(
    mean_space = mean(space_m3),
    sd_space = sd(space_m3),
    min_space = min(space_m3),
    max_space = max(space_m3),
    median_space = median(space_m3),

    mean_stock = mean(stock_qty),
    sd_stock = sd(stock_qty),
    min_stock = min(stock_qty),
    max_stock = max(stock_qty),
    median_stock = median(stock_qty)
  )

cat("\n==== CATEGORY-LEVEL SUMMARY ====\n")
print(cat_summary)
```

```
# ---------------------------------------------------------
# 3. Category Share of Total Space
# ---------------------------------------------------------

cat_share <- cat_daily %>%
  group_by(prod_cat) %>%
  summarise(total_space = sum(space_m3)) %>%
  mutate(share = total_space / sum(total_space))

cat("\n==== CATEGORY SHARE OF TOTAL SPACE ====\n")
print(cat_share)
```

```
# ---------------------------------------------------------
# 4. Create wide category space matrix for correlation + VAR
# ---------------------------------------------------------

# cat_space_wide <- cat_daily %>%
#   select(stock_xfer_date, prod_cat, space_m3) %>%
#   pivot_wider(names_from = prod_cat,
#               values_from = space_m3,
#               values_fill = 0) %>%
#   arrange(stock_xfer_date)
#
# cat("\n==== CATEGORY SPACE (WIDE FORMAT) ====\n")
# print(head(cat_space_wide))
```

```
# --------------------------------------------------------
# 5. Correlation Across Categories (space)
# --------------------------------------------------------

cor_cat_space <- cor(cat_space_wide[, -1])
cat("\n==== CATEGORY SPACE CORRELATION MATRIX ====\n")
print(round(cor_cat_space, 3))
```

```
# --------------------------------------------------------
# 6. Plot: Daily Category Space Usage (m³)
# --------------------------------------------------------

ggplot(cat_daily, aes(x = stock_xfer_date, y = space_m3, color = prod_cat)) +
  geom_line(linewidth = 0.8) +
  labs(
    title = "Daily Space Usage by Product Category",
    x = "Date",
    y = "Space Used (m³)",
    color = "Product Category"
  ) +
  theme_minimal()
```

```
# --------------------------------------------------------
# 7. Plot: Category Space Distribution
# --------------------------------------------------------

ggplot(cat_daily, aes(x = prod_cat, y = space_m3, fill = prod_cat)) +
  geom_boxplot(alpha = 0.8) +
  labs(
    title = "Distribution of Daily Space Usage by Category",
    x = "Category",
    y = "Space (m³)"
  ) +
  theme_minimal() +
  theme(legend.position = "none")
```

```r
# ----------------------------------------------------------
# 8. Plot: Daily Category Stock Transfers
# ----------------------------------------------------------

ggplot(cat_daily, aes(x = stock_xfer_date, y = stock_qty, color = prod_cat)) +
  geom_line(linewidth = 0.8) +
  labs(
    title = "Daily Stock Transfers by Product Category",
    x = "Date",
    y = "Units Transferred"
  ) +
  theme_minimal()
```

```r
# ----------------------------------------------------------
# 9. Category-level summary table for the report
# ----------------------------------------------------------

category_summary_table <- cat_summary %>%
  mutate(
    avg_daily_share = cat_share$share[match(prod_cat, cat_share$prod_cat)]
  )

cat("\n==== FINAL CATEGORY SUMMARY (SPACE + STOCK + SHARE) ====\n")
print(category_summary_table)
```

```r
#############################################################
# STATIONARITY RESULTS FOR TOTAL SPACE + CATEGORY SERIES
#############################################################

library(tseries)
library(forecast)
library(ggplot2)
```

```r
# ----------------------------------------------------------
# 1. ADF Test for Total Space Used (m³)
# ----------------------------------------------------------

cat("==== ADF Test: Total Space (Level) ====\n")
adf_level <- adf.test(ts_total_space)
print(adf_level)

cat("\n==== ADF Test: Total Space (First Difference) ====\n")
```

```r
adf_diff <- adf.test(diff(ts_total_space))
print(adf_diff)
```

```r
# ----------------------------------------------------------
# 2. ACF & PACF Plots for Total Space Used
# ----------------------------------------------------------

par(mfrow = c(2, 1))

acf(ts_total_space, main = "ACF - Total Space (Level)")
pacf(ts_total_space, main = "PACF - Total Space (Level)")

par(mfrow = c(2, 1))

acf(diff(ts_total_space), main = "ACF - Total Space (First Difference)")
pacf(diff(ts_total_space), main = "PACF - Total Space (First Difference)")

par(mfrow = c(1, 1))
```

```r
# ----------------------------------------------------------
# 3. ADF Tests for Category-Level Series
# ----------------------------------------------------------

cat("\n==== ADF Tests: Category Space (Level Series) ====\n")

cat_names <- colnames(ts_cat_space)

for (i in 1:ncol(ts_cat_space)) {
  cat("\nCategory:", cat_names[i], "\n")
  print(adf.test(ts_cat_space[, i]))
}

cat("\n==== ADF Tests: Category Space (First Differences) ====\n")

for (i in 1:ncol(ts_cat_space)) {
  cat("\nCategory:", cat_names[i], " (Differenced)\n")
  print(adf.test(diff(ts_cat_space[, i])))
}
```

```r
# ----------------------------------------------------------
# 4. ACF & PACF Plots for Category-Level Series
# ----------------------------------------------------------
```

```
par(mfrow = c(2, 2))

for (i in 1:ncol(ts_cat_space)) {
  acf(ts_cat_space[, i],
      main = paste("ACF -", cat_names[i], "(Level)"))
}

par(mfrow = c(2, 2))

for (i in 1:ncol(ts_cat_space)) {
  acf(diff(ts_cat_space[, i]),
      main = paste("ACF -", cat_names[i], "(First Difference)"))
}

par(mfrow = c(2, 2))

for (i in 1:ncol(ts_cat_space)) {
  pacf(ts_cat_space[, i],
       main = paste("PACF -", cat_names[i], "(Level)"))
}

par(mfrow = c(2, 2))

for (i in 1:ncol(ts_cat_space)) {
  pacf(diff(ts_cat_space[, i]),
       main = paste("PACF -", cat_names[i], "(First Difference)"))
}

par(mfrow = c(1, 1))
```

```
acf(ts_cat_space)
```

```
############################################################
#   FORECASTED RESULTS FOR VAR MODEL
############################################################

library(vars)
library(tidyverse)
```

```r
# -------------------------------
# 1. Forecast 30 days ahead
# -------------------------------
var_fc <- predict(var_model, n.ahead = 30)

# View raw forecast structure
print(var_fc)
```

```r
# -------------------------------
# 2. Extract forecasted changes (differenced predictions)
# -------------------------------
# var_fc$fcst is a list with one entry per category
# Each contains mean, lower, upper forecasts

fc_means <- sapply(var_fc$fcst, function(x) x[, "fcst"])
colnames(fc_means) <- names(var_fc$fcst)

cat("=== Forecasted FIRST DIFFERENCES (VAR) ===\n")
print(round(fc_means, 4))
```

```r
# -------------------------------
# 3. Convert differenced forecasts back to actual LEVELS
# -------------------------------
# Last observed category space values from 2023
last_obs <- tail(cat_space_wide[, -1], 1)

# Cumulative sum of differenced forecasts + last observed values
fc_levels <- apply(fc_means, 2, cumsum) %>%
  sweep(1, as.numeric(last_obs), `+`)

fc_levels <- as.data.frame(fc_levels)
fc_levels$Day <- 1:nrow(fc_levels)

cat("\n=== Forecasted CATEGORY SPACE LEVELS (m³) ===\n")
print(round(fc_levels, 4))
```

```r
# -------------------------------
# 4. Optional: plot VAR forecasts
# -------------------------------
fc_long <- fc_levels %>%
  pivot_longer(cols = -Day, names_to = "Category", values_to = "Forecast_m3")
```

```r
ggplot(fc_long, aes(x = Day, y = Forecast_m3, color = Category)) +
  geom_line(size = 1.2) +
  theme_minimal() +
  ggtitle("VAR Forecast - Category-Level Space Usage (m³)") +
  labs(x = "Forecast Day", y = "Space Used (m³)") +
  theme(
    plot.title = element_text(size = 14, face = "bold"),
    legend.title = element_blank()
  )
```

```r
############################################################
# COMPLETE EXPLORATORY DATA ANALYSIS (EDA)
# Dataset: products_02.csv
############################################################

library(tidyverse)
library(lubridate)
library(psych)
library(GGally)
library(gridExtra)
```

```r
# ---------------------------------------------------------
# 1. Load Data & Initial Inspection
# ---------------------------------------------------------
raw <- read_csv("products_02.csv")

cat("==== HEAD OF DATA ====\n")
print(head(raw))

cat("\n==== STRUCTURE OF DATA ====\n")
str(raw)

cat("\n==== SUMMARY OF RAW DATA ====\n")
summary(raw)

cat("\n==== CHECK FOR MISSING VALUES ====\n")
print(colSums(is.na(raw)))
```

```r
# ---------------------------------------------------------
# 2. Clean & Prepare Variables
# ---------------------------------------------------------
```

```
raw <- raw %>%
  mutate(
    # stock_xfer_date = ymd(stock_xfer_date),
    prod_dimensions = str_remove(prod_dimensions, " cm")
  ) %>%
  separate(prod_dimensions, into = c("len_cm", "width_cm", "height_cm"),
           sep = "x", convert = TRUE) %>%
  mutate(
    volume_cm3 = len_cm * width_cm * height_cm,
    space_used_cm3 = volume_cm3 * stock_qty,
    space_used_m3 = space_used_cm3 / 1e6
  )

cat("\n==== AFTER CLEANING ====\n")
summary(raw)
```

```
# ---------------------------------------------------------
# 3. Numerical Summary Statistics
# ---------------------------------------------------------
num_vars <- raw %>%
  select(stock_qty, len_cm, width_cm, height_cm, volume_cm3, space_used_m3)

cat("\n==== SUMMARY STATISTICS (NUMERIC VARIABLES) ====\n")
print(describe(num_vars))
```

```
# ---------------------------------------------------------
# 4. Category-Level Summaries
# ---------------------------------------------------------
cat("\n==== CATEGORY SUMMARY ====\n")
cat_summary <- raw %>%
  group_by(prod_cat) %>%
  summarise(
    n = n(),
    total_stock = sum(stock_qty),
    avg_stock = mean(stock_qty),
    total_space_m3 = sum(space_used_m3),
    avg_space_m3 = mean(space_used_m3),
    avg_volume_cm3 = mean(volume_cm3)
  )

print(cat_summary)
```

```
num_vars <- raw %>%
  select(stock_qty, len_cm, width_cm, height_cm, volume_cm3, space_used_m3)

describe(num_vars)
```

```
# ---------------------------------------------------------
# 5. Visual Distributions - Stock Transfers
# ---------------------------------------------------------
p1 <- ggplot(raw, aes(stock_qty)) +
  geom_histogram(fill="steelblue", color="black", bins=40) +
  theme_minimal() + ggtitle("Histogram: Stock Transfers")

p2 <- ggplot(raw, aes(y = stock_qty, x = prod_cat, fill = prod_cat)) +
  geom_boxplot() +
  theme_minimal() + ggtitle("Stock Transfers by Product Category")

grid.arrange(p1, p2, ncol = 2)
```

```
# ---------------------------------------------------------
# 6. Visual Distributions - Space Used
# ---------------------------------------------------------
p3 <- ggplot(raw, aes(space_used_m3)) +
  geom_histogram(fill="darkgreen", color="black", bins=40) +
  theme_minimal() + ggtitle("Histogram: Space Used (m3)")

p4 <- ggplot(raw, aes(x = prod_cat, y = space_used_m3, fill = prod_cat)) +
  geom_boxplot() +
  theme_minimal() + ggtitle("Space Used by Category (m3)")

grid.arrange(p3, p4, ncol = 2)
```

```
# ---------------------------------------------------------
# 7. Time Series Aggregation
# ---------------------------------------------------------
daily <- raw %>%
  group_by(stock_xfer_date) %>%
  summarise(
    total_stock = sum(stock_qty),
    total_space_m3 = sum(space_used_m3)
  )
```

```r
cat("\n==== DAILY SUMMARY ====\n")
print(head(daily))

# Time series plots
p5 <- ggplot(daily, aes(stock_xfer_date, total_stock)) +
  geom_line(color="steelblue") +
  theme_minimal() + ggtitle("Daily Total Stock Transfers")

p6 <- ggplot(daily, aes(stock_xfer_date, total_space_m3)) +
  geom_line(color="darkgreen") +
  theme_minimal() + ggtitle("Daily Total Space Used (m3)")

grid.arrange(p5, p6, ncol = 1)
```

```r
# ---------------------------------------------------------
# 8. Category-Level Time Series
# ---------------------------------------------------------
cat_daily <- raw %>%
  group_by(stock_xfer_date, prod_cat) %>%
  summarise(space_m3 = sum(space_used_m3), stock_qty = sum(stock_qty),
            .groups = "drop")

p7 <- ggplot(cat_daily, aes(stock_xfer_date, space_m3, color = prod_cat)) +
  geom_line() +
  theme_minimal() +
  ggtitle("Daily Space Used by Category")

p8 <- ggplot(cat_daily, aes(stock_xfer_date, stock_qty, color = prod_cat)) +
  geom_line() +
  theme_minimal() +
  ggtitle("Daily Stock Transfers by Category")

grid.arrange(p7, p8, ncol = 1)
```

```r
# ---------------------------------------------------------
# 9. Correlation Analysis
# ---------------------------------------------------------
cor_vars <- raw %>%
  select(stock_qty, len_cm, width_cm, height_cm, volume_cm3, space_used_m3)

cat("\n==== CORRELATION MATRIX ====\n")
print(cor(cor_vars))
```

```
ggpairs(cor_vars) + ggtitle("Pairwise Relationships")
```

```
# ---------------------------------------------------------
# 10. Distribution of Item Dimensions & Volume
# ---------------------------------------------------------
p9 <- ggplot(raw, aes(volume_cm3)) +
  geom_histogram(fill="purple", color="black", bins=40) +
  theme_minimal() + ggtitle("Histogram: Item Volume (cm3)")

p10 <- ggplot(raw, aes(x = prod_cat, y = volume_cm3, fill = prod_cat)) +
  geom_boxplot() +
  theme_minimal() +
  ggtitle("Volume Distribution by Product Category")

grid.arrange(p9, p10, ncol = 2)
```

```
# ---------------------------------------------------------
# 11. Scatterplots (Stock vs Space, Volume vs Space)
# ---------------------------------------------------------
p11 <- ggplot(raw, aes(stock_qty, space_used_m3)) +
  geom_point(alpha=0.3, color="blue") +
  theme_minimal() +
  ggtitle("Stock Transfers vs Space Used (m3)")

p12 <- ggplot(raw, aes(volume_cm3, space_used_m3)) +
  geom_point(alpha=0.3, color="red") +
  theme_minimal() +
  ggtitle("Volume (cm3) vs Space Used (m3)")

grid.arrange(p11, p12, ncol = 2)

############################################################
# END OF COMPLETE EDA SCRIPT
############################################################
```