**COMP1204 Data Management**

**Coursework-2 LINUX**

**Group Members:**

|  |  |  |
| --- | --- | --- |
| **No.** | **Name** | **University Email** |
| 1. | Brandon Ting Wee Kang | bwkt1n22@soton.ac.uk |
| 2. | Pang Jia Hui | jhp1n22@soton.ac.uk |

**Submitted Documents in ZIP File:**

|  |  |  |
| --- | --- | --- |
| **No.** | **Documents** | **File Name** |
| 1. | Report | DM CW2 – Report |
| 2. | Scraper/Tracker Script: | price\_tracker.sh |
| 3. | Database | pricetracker.sql |
| 4. | Github Repository | <https://github.com/brandon-nx/OnlinePromotionStoreTracker> |

Website URL:

<https://www.publicpackaging.com/>

Products URL:

* "https://www.jsshoppu.com/showproducts/productid/4312336/cid/448063/laundry-cleaner-detergent-paper-tablet-pembersih-baju-%E6%B4%97%E8%A1%A3%E7%BA%B8-%E6%B3%A1%E6%B3%A1%E7%BA%B8/"
* "https://www.jsshoppu.com/showproducts/productid/4312047/cid/448063/seamless-wall-hanging-mop-hook-bathroom-broom-hanger-%E5%BC%BA%E5%8A%9B%E6%97%A0%E7%97%95%E6%8B%96%E6%8A%8A%E5%A4%B9-%E5%85%8D%E6%89%93%E5%AD%94%E6%8C%82%E6%89%AB%E6%8A%8A%E6%9E%B6-1-pc/
* "https://www.jsshoppu.com/showproducts/productid/4308930/cid/448063/simple-towel-hanger-towel-holder-pemegang-tuala-rak-tuala-%E6%AF%9B%E5%B7%BE%E6%9E%B6/"
* "https://www.jsshoppu.com/showproducts/productid/4308972/cid/448063/creative-2l-gradient-frosted-colorful-scale-straw-portable-water-bottle-%E6%B8%90%E5%8F%98%E8%89%B2%E7%A3%A8%E7%A0%82%E7%82%AB%E5%BD%A9%E5%88%BB%E5%BA%A6%E5%90%B8%E7%AE%A1%E4%BE%BF%E6%90%BA%E6%B0%B4%E7%93%B6/
* "https://www.jsshoppu.com/showproducts/productid/4312065/cid/448063/super-clean-gel-compound-cleaning-gel-jelly-dust-cleaning-80g%E5%8D%A4/"

My own refer: (don’t delete 😊)

cd /mnt/c/Users/Hp/Desktop/OneDrive\ -\ University\ of\ Southampton/Projects/Others/OnlinePromotionStoreTracker/scripts

# 1.0 Introduction

The project is an online store promotion tracker that monitors key data such as pricing, rating, stock availability and number of items sold for products listed on the Lazada platform. By collecting this data regularly and tracking changes over time, the tracker intends to assist customers in making more informed purchasing decisions. In today's dynamic online business, where prices and promotions change quickly, automated tracking of online store promotions has various advantages over manual monitoring. It allows consumers to identify price trends and promotions easily. Furthermore, the tracker simplifies the decision-making process for customers, allowing them to make quick decisions and act promptly on opportunities to save money.

The project collects data using Unix scripts to ensure that it is collected within a particular period and stored in a MySQL database. Once enough data has been collected, the next step is to produce graphs to visualise trends and analyse patterns in the data, allowing users to track data and make quick decisions about purchasing things on an online platform.   
Overall, the online store promotion tracker is an effective tool for consumers seeking to optimize their online shopping experience by staying up to date with special offers, price variations, and stock reminders on the Lazada platform. The tracker enables customers to make better purchase decisions and save more money by providing them with fast and accurate information.

# 2.0 Table of Content

# 3.0 Timeline

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Phase** | **Process** | **Week1** | | | | | | | **Week2** | | | | | | | **Week3** |
| **Date** | **29/4** | **30/4** | **1/5** | **2/5** | **3/5** | **4/5** | **5/5** | **6/5** | **7/5** | **8/5** | **9/5** | **10/5** | **11/5** | **12/5** | **13/5** |
| **Database Design** | Design the Schema |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Create Database |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Scraper/ Tracker Script** | Fetching Web Data |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Parsing Data |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Data Manipulation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Insert into Database |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Crontab Setup** | Scheduling |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Plotting Script** | Create script |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Plot Types |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Report Writing** | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Github Repository** | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Report Submission** | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# 4.0 Database Design

**Jiahui insert the database screenshot**

(whole table screenshot)

(1s table)

(2nd table)

(3rd table)

(4th table)

.

.

.

# 5.0 Scraper/Tracker Script

**5.1 Fetching Web Data**

PRODUCT\_URLS=(

    "https://www.jsshoppu.com/showproducts/productid/4312336/cid/448063/laundry-cleaner-detergent-paper-tablet-pembersih-baju-%E6%B4%97%E8%A1%A3%E7%BA%B8-%E6%B3%A1%E6%B3%A1%E7%BA%B8/",

    "https://www.jsshoppu.com/showproducts/productid/4312047/cid/448063/seamless-wall-hanging-mop-hook-bathroom-broom-hanger-%E5%BC%BA%E5%8A%9B%E6%97%A0%E7%97%95%E6%8B%96%E6%8A%8A%E5%A4%B9-%E5%85%8D%E6%89%93%E5%AD%94%E6%8C%82%E6%89%AB%E6%8A%8A%E6%9E%B6-1-pc/",

    "https://www.jsshoppu.com/showproducts/productid/4308930/cid/448063/simple-towel-hanger-towel-holder-pemegang-tuala-rak-tuala-%E6%AF%9B%E5%B7%BE%E6%9E%B6/",

    "https://www.jsshoppu.com/showproducts/productid/4308972/cid/448063/creative-2l-gradient-frosted-colorful-scale-straw-portable-water-bottle-%E6%B8%90%E5%8F%98%E8%89%B2%E7%A3%A8%E7%A0%82%E7%82%AB%E5%BD%A9%E5%88%BB%E5%BA%A6%E5%90%B8%E7%AE%A1%E4%BE%BF%E6%90%BA%E6%B0%B4%E7%93%B6/",

    "https://www.jsshoppu.com/showproducts/productid/4312065/cid/448063/super-clean-gel-compound-cleaning-gel-jelly-dust-cleaning-80g%E5%8D%A4/"

)

for URL in "${PRODUCT\_URLS[@]}"; do

    PRODUCT\_NAME=$(echo "$URL" | awk -F '/' '{print $(NF-1)}' | cut -c1-30)

    OUTPUT\_FILE="${PRODUCT\_NAME}\_page.html"

    curl -s "$URL" -o "$OUTPUT\_FILE"

    # Check if curl succeeded

    if [ $? -ne 0 ]; then

        echo "Failed to fetch data from $URL"

        continue

    fi

    echo "Web data successfully fetched and saved to $OUTPUT\_FILE"

    parseData "$OUTPUT\_FILE" "$PRODUCT\_NAME" "$URL"

done

**5.2 Parsing Data:** Use tools like **grep** and **awk** to extract relevant information from the HTML content.

parseData() {

    local file="$1"

    local product\_name="$2"

    local product\_url="$3"

    # Parse price

    price=$(grep -oP 'product:price:amount" content="\K[\d.]+' "$file")

    # Parse stock

    stock=$(awk 'BEGIN{RS="<"; FS=">"; IGNORECASE=1} /class="product\_qty\_availble"/ && !found {print $2; found=1}' "$file" | grep -oP '\d+' | head -n 1 | tr -d '\n')

    # Parse category

    category=$(grep -oP 'property="product:category" content="\K[^"]+' "$file")

    echo "Parsed Data: Product: $product\_name, Price: RM$price, Stock: $stock, Category: $category"

    dataManipulation "$product\_name" "$price" "$stock" "$category" "$product\_url"

}

**5.3 Data Manipulation:** Process and convert extracted data into appropriate formats

dataManipulation() {

    local product\_name="$1"

    local price="$2"

    local stock="$3"

    local category="$4"

    local product\_url="$5"

    # Convert price to a float if it's not already

    price=$(printf "%.2f" "$price")

    # Ensure stock is an integer

    stock=$(printf "%d" "$stock")

    echo "Manipulated Data: Product: $product\_name, Price: RM$price, Stock: $stock, Category: $category"

    insertIntoDatabase "$product\_name" "$price" "$stock" "$category" "$product\_url"

}

**5.4 Insert into Database:** Add MySQL commands to insert collected data into the database, ensuring error handling for database interactions.

insert\_data() {

    table=$1

    columns=$2

    values=$3

    # MySQL command to insert data into the specified table

    echo "Inserting data into the table $table"

    query="INSERT INTO $table ($columns) VALUES ($values);"

    error=$(mysql -u"$DB\_USER" -p"$DB\_PASS" -D"$DB\_NAME" -e "$query" 2>&1 > /dev/null)

    if [ ! -z "$error" ]; then

        echo "Error inserting into $table: $error"

    else

        echo "Data inserted successfully into $table."

    fi

}

insertIntoDatabase() {

    product\_name="$1"

    price="$2"

    stock="$3"

    category="$4"

    product\_url="$5"

    # Get the current datetime

    current\_datetime=$(date '+%Y-%m-%d %H:%M:%S')

    # Generate new product ID

    new\_product\_id=$(generateProductID)

    new\_details\_id=$(generateDetailsID)

    new\_track\_id=$(generateTrackID)

    # Insert data into the products table

    insert\_data "products" "productID, productName, category, URL" "'$new\_product\_id', '$product\_name', '$category', '$product\_url'"

    # Insert data into the trackdetails table

    insert\_data "trackdetails" "trackID, price, stock, dateCollected" "'$new\_track\_id', $price, $stock, '$current\_datetime'"

    # Insert data into the productdetails table

    insert\_data "productdetails" "detailsID, productID, trackID" "'$new\_details\_id', '$new\_product\_id', '$new\_track\_id'"

}

# 6.0 Contab Setup and Error Handling

* + **Scheduling:** Set up a crontab entry to run the scraper script at regular intervals, such as daily or hourly, to ensure continuous data collection.

6.1 Crontab setup (10%)

6.2 Error handling, such as what if network is down or the website itself is down while the script is running, or what if the website blocks your script (10%)

# 7.0 Plotting Script

* + **Create the Script:** Write a shell script to generate at least 10 different plots using GNU Plot or similar tools.
  + **Plot Types:** Implement plotting functions that show trends such as:
    - Price changes over time.
    - Stock changes over time.
    - Rating changes over time.
  + **Script Organization:** Ensure the plotting script is organized into easily callable functions.

# 8.0 Conclusion

To summarise, the online store promotion tracker developed for the Lazada platform represents a significant development in the field of online shopping platforms. The tracker is an invaluable tool for consumers intending to make informed purchasing decisions because it consistently collects and analyses key data such as pricing, rating, stock availability, and the number of things sold. Throughout this project, we have demonstrated how automated tracking can provide consumers with fast and accurate information on product promotions and pricing patterns. By eliminating the need for manual data collection and analysis, the tracker simplifies consumer decision-making, allowing them to move quickly on opportunities to save money and take advantage of promotional offers. The project also further proved the reliability of Unix scripts for data collection and storage.

By using Unix scripts to collect data periodically and store it in a MySQL database, we ensure that users have access to the most current information when making purchasing decisions. Furthermore, the implementation of data visualisation techniques like graph plotting enables users to monitor trends pricing patterns and stock availability, allowing them to make intelligent decisions about when to buy, sell, or wait for great deals. Looking ahead, there are various chances for future improvements and expansions to the online store promotion tracker. This involves enhancing data collection techniques, providing support for new online tools, and including features like email notifications for price decreases and stock updates.

By continuing to innovate and improve the existing structure, we can increase the tracker's utility and efficiency in supporting consumers with their online shopping needs. In conclusion, the online store promotion tracker is a useful tool for consumers wishing to improve their online shopping experience by staying updated about special offers, pricing variations, and stock availability on the Lazada platform. By providing customers with fast and accurate information, the tracker enables them to make better purchase decisions and save more money.

# 9.0 Appendices

Jiahui add your database sketch

Reference

fake\_product\_page.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Product Page</title>

</head>

<body>

    <div class="product-details">

        <span class="price" data-price="1300.50">RM1300.50898</span>

        <div class="stock-info" data-stock="20">20 units available.</div>

        <div class="ratings" data-rating="4.7">Rated 4.7 out of 5 stars</div>

    </div>

</body>

</html>

test\_parseDataAndDataManipulation.sh

#!/bin/bash

# Function to parse data

parseData() {

    local file="$1"

    local product\_name="$2"

    # Parse price from the new format

    price=$(grep -oP 'class="price" data-price="\K[\d.]+' "$file")

    # Parse stock from the new format

    stock=$(grep -oP 'class="stock-info" data-stock="\K[\d.]+' "$file")

    # Parse rating from the new format

    rating=$(grep -oP 'class="ratings" data-rating="\K[\d.]+' "$file")

    echo "Parsed Data: Price: RM$price, Stock: $stock units, Rating: $rating stars"

    dataManipulation "$product\_name" "$price" "$stock" "$rating"

}

# Function to manipulate data

dataManipulation() {

    local product\_name="$1"

    local price="$2"

    local stock="$3"

    local rating="$4"

    # Convert price to a float

    price=$(printf "%.2f" "$price")

    # Ensure stock is an integer

    stock=$(printf "%d" "$stock")

    # Convert rating to one decimal place

    rating=$(printf "%.1f" "$rating")

    echo "Manipulated Data: Product: $product\_name, Price: RM$price, Stock: $stock units, Rating: $rating stars"

}

# Testing the parseData function with the new HTML file format

echo "Testing parseData function..."

parseData "fake\_product\_page.html" "Smartphone Galaxy S22 Ultra"