# Assignment 1

# Web Scraping Report

## Website Information

### Koodo Mobile

**Website**: https://www.koodomobile.com/en

Koodo Mobile is a Canadian mobile flanker brand started by Telus in 2008. This website offers various mobile phones and plans to its customers. Koodo is known for providing affordable and flexible mobile services without fixed-term contracts, appealing to a wide range of customers looking for reliable yet budget-friendly options.

## Task 1 :

**1. Opening the Website:**
I initialized the Selenium WebDriver for Chrome and opened the Koodo Mobile website.

**2. Interacting with Elements:**
I located and interacted with the dropdown menu to navigate to the "Phones" section using Selenium commands.

**3. Extracting Data:**
I extracted phone details such as names, rates, and images from the webpage using the findElements and findElement methods provided by Selenium.

**4. Saving Data:**
I saved the extracted data into a CSV file using the FileWriter class.

In this image, I am setting up WebDriver and utilizing the command 'driver.get' to navigate to the Koodo Mobile homepage. Afterward, I'm waiting for elements to load, interacting with the dropdown menu, and finally extracting phone details.

```
public class Main {
    public static void main(String[] args) throws IOException {
        // Set the path for the ChromeDriver executable
        System.setProperty("webdriver.chrome.driver", "C:/chromedriver/chromedriver.exe");

        // Initialize a new WebDriver instance using ChromeDriver
        WebDriver driver = new ChromeDriver();

        // Navigate to the Koodo Mobile website
        driver.get("https://www.koodomobile.com/en");
        WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));

        // Define the path for the output CSV file and delete it if it already exists
        Path file = Paths.get( first: "C:/Users/sanja/OneDrive/Desktop/Koodo.csv");
        Files.deleteIfExists(file);
        System.out.println("File deleted successfully.");

        // Interact with the dropdown menu to navigate to the Phones section
        WebElement DropdownPhone = wait.until(ExpectedConditions.visibilityOfElementLocated(By.cssSelector(".header__menu-item-button")));
        DropdownPhone.click();
        WebElement phone = driver.findElement(By.cssSelector("li[data-test='subMenuItem0']"));
        phone.click();

        // Extract phone details
        List<WebElement> phones = driver.findElements(By.cssSelector(".sc-c64e8282-0.ilWKoL"));

        String filePath = "C:/Users/sanja/OneDrive/Desktop/Koodo.csv";
        try (FileWriter fileWriter = new FileWriter(filePath, append: true)) {
            fileWriter.append("Phones, Rates, Images\n\n");
```

Figure 1 : opening the web browser using Selenium.

## Task 2 :

### 1. Interacting with the Dropdown Menu:

I used Selenium commands to click on the dropdown menu and navigate to both the "Phones" and "Plans" sections under the "Shop" category.

### 2. Extracting Data from Multiple Pages:

I extracted both phone and plan details from their respective pages and appended the data to the same CSV file to combine the results.

```
        // Extract phone details
        List<WebElement> phones = driver.findElements(By.cssSelector(".sc-c64e8282-0.ilWKoL"));

        String filePath = "C:/Users/sanja/OneDrive/Desktop/Koodo.csv";
        try (FileWriter fileWriter = new FileWriter(filePath, append: true)) {
            fileWriter.append("Phones, Rates, Images\n\n");

            for (WebElement phoneitem : phones) {

                String phonename = phoneitem.findElement(By.cssSelector("[class='css-1ryng56']")).getText().replaceAll( regex: "\\s+", replacement: " ").trim();
                String model = phoneitem.findElement(By.cssSelector("[data-media='rnmq-265e82d3 rnmq-8ac0ab7a rnmq-2568a35c rnmq-fcb13a46 rnmq-6d6ae753']")).get
                String rate = phoneitem.findElement(By.cssSelector(".PriceLockup__PriceTextContainer-components-web__sc-1x6duay-6.gIneyq")).getText().replaceAll
                String imageUrl = phoneitem.findElement(By.cssSelector("picture")).findElement(By.cssSelector("img")).getAttribute( name: "src");

                // Write phone details to the CSV file
                fileWriter.write( str: phonename + " " + model + "," + rate + "," + imageUrl + "\n");
            }
        }
    }
}
```

Figure 2 : Extracting Data from "Phones" page.

1. **phonename**: The name of the phone.
2. **model**: The model of the phone.
3. **rate**: The rate of the phone.
4. **imageUrl**: The URL of the image of the phone.

```java
try (FileWriter fileWriter = new FileWriter(filePath, append: true)) {

    // Extract plan details
    List<WebElement> plans = driver.findElements(By.cssSelector(".sc-cPiKLX.bWWNcz"));
    fileWriter.append("\n\n\n\n\n\nPrices, Plans, Details\n\n");
    for (WebElement planitem : plans) {
        // Get the plan name
        String prices = planitem.findElement(By.cssSelector(".PriceLockup__PriceLockupContainer-components-web__sc-1x6duay-0.fqMdJw")).getText().replaceAll(
        String data = planitem.findElement(By.cssSelector(".sc-eDPEul.sc-eldPxv.ejyAOJ.iZsFvQ")).findElement(By.cssSelector(".css-175oi2r.r-eqz5dr")).getTex
        String details = planitem.findElement(By.cssSelector(".Paragraph__StyledParagraph-components-web__sc-1bg9r8p-0.dfAHHh")).getText().replaceAll( regex:

        // Write plan details to the CSV file
        if (!prices.equals("") || !data.equals("") || !details.equals("")) {
            fileWriter.write( str: prices + "," + data + "," + details + "\n");
        }

    }

}
```

Figure 3 : Extracting Data from "Plans" page.

1. **prices**: The prices of the plan.
2. **data**: The data details of the plan.
3. **details**: Other details of the plan.

```java
String filePath = "C:/Users/sanja/OneDrive/Desktop/Koodo.csv";
try (FileWriter fileWriter = new FileWriter(filePath, append: true)) {
    fileWriter.append("Phones, Rates, Images\n\n");

    for (WebElement phoneitem : phones) {

        String phonename = phoneitem.findElement(By.cssSelector("[class='css-1ryng56']")).getText().replaceAll( regex: "\\s+", replacement: " ").trim();
        String model = phoneitem.findElement(By.cssSelector("[data-media='rnmq-265e82d3 rnmq-8ac0ab7a rnmq-2568a35c rnmq-fcb13a46 rnmq-6d6ae753']")).get
        String rate = phoneitem.findElement(By.cssSelector(".PriceLockup__PriceTextContainer-components-web__sc-1x6duay-6.gIneyq")).getText().replaceAll
        String imageUrl = phoneitem.findElement(By.cssSelector("picture")).findElement(By.cssSelector("img")).getAttribute( name: "src");

        // Write phone details to the CSV file
        fileWriter.write( str: phonename + " " + model + "," + rate + "," + imageUrl + "\n");
```

Figure 4 : FileWriter

In my script, I utilized 'FileWriter' to manage writing operations to a CSV file. Firstly, a FileWriter object ('fileWriter') is created by opening a file for writing, with the file path specified as 'filePath'. Subsequently, the 'append()' method is employed to add data to the file, while 'write()' is utilized to write each line of data. Finally, the file writer is closed using 'close()' .

# Task 3 :

**Waiting for Elements to Load:**

I used the `WebDriverWait` class in Selenium to dynamically wait for elements to reach a specific state before proceeding. This is crucial for handling dynamic content or delayed loading. In my code, I set a 10-second timeout to wait for the dropdown menu button's visibility. This duration ensures flexibility in handling varying loading times. I used `ExpectedConditions`, which offers various conditions like `visibilityOfElementLocated`, to wait for an element to be visible. This ensures reliable interaction with fully loaded elements. By using `WebDriverWait` with `ExpectedConditions`, my script waits for elements to fully load before interacting with them. This makes the scraping process more reliable.



```java
driver.get("https://www.koodomobile.com/en");
WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));

// Define the path for the output CSV file and delete it if it already exists
Path file = Paths.get( first: "C:/Users/sanja/OneDrive/Desktop/Koodo.csv");
Files.deleteIfExists(file);
System.out.println("File deleted successfully.");

// Interact with the dropdown menu to navigate to the Phones section
WebElement DropdownPhone = wait.until(ExpectedConditions.visibilityOfElementLocated(By.cssSelector(".header__menu-item-button")));
DropdownPhone.click();
```

Figure 5 : Waiting for elements to load

# Outputs:



Figure 6 : output of page 1

```
Price : $ 30 .00 per month
Data : Pay-per-use data
Details : Unlimited Canada-wide minutes
Price : $ 34 .00 per month
Data : 20 GB 4G speed
Details : Pick 1 Free Perk from options like Premium Voicemail
Price : $ 39 .00 per month
Data : 40 GB 4G speed
Details : Pick 1 Free Perk from options like Rollover Data
Price : $ 44 .00 per month
Data : 50 GB 4G speed
Details : Pick 1 Free Perk from options like Unlimited Long Distance Pack
Price : $ 15 .00 per month
Data : 250 MB 3G speed
Details : 100 Canada-wide minutes with unlimited incoming minutes
Price : $ 35 .00 per month
Data : 3 GB 3G speed
Details : Unlimited Canada-wide minutes
```

Figure 7 : Output of page 2



Figure 8 : CSV file of page 1

Figure 9 : CSV file of page 2

In the outputs image, you can see the phone details from the "Phones" page written in the "Phone Names," "Rates," and "Image URL" columns. The plans from the "Plans" page are written in the "Prices," "Plans," and "Details" columns. Data scraped from both the "Phones" and "Plans" pages is stored in the same CSV file.
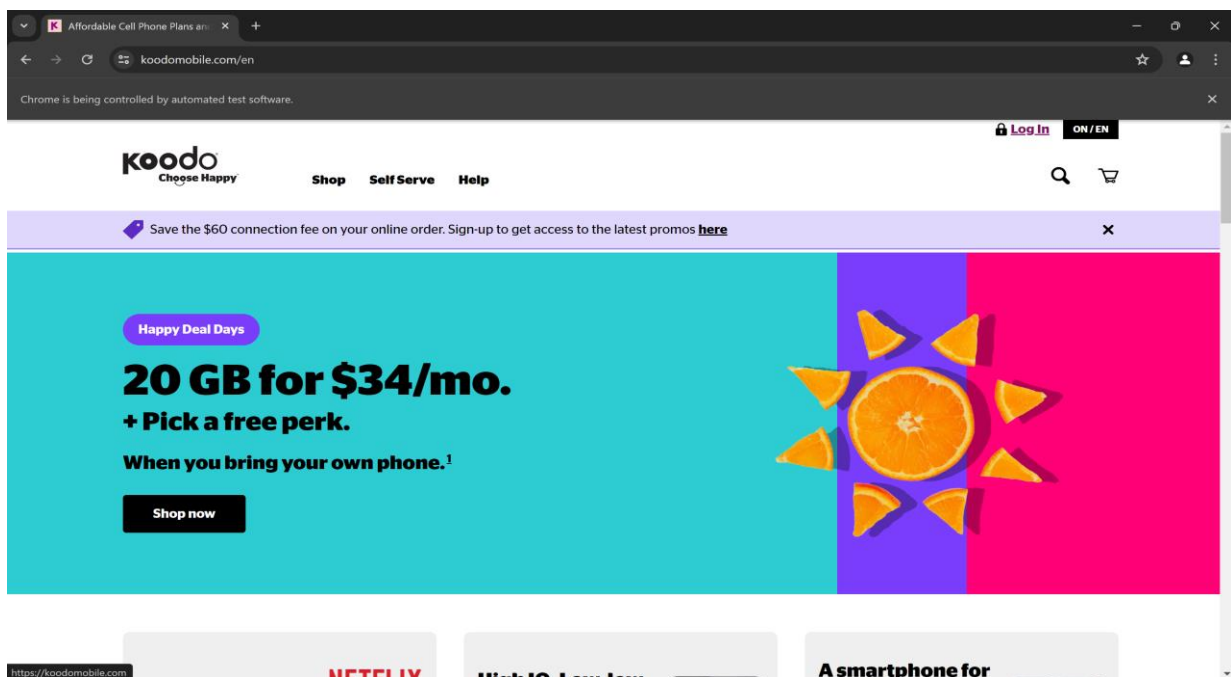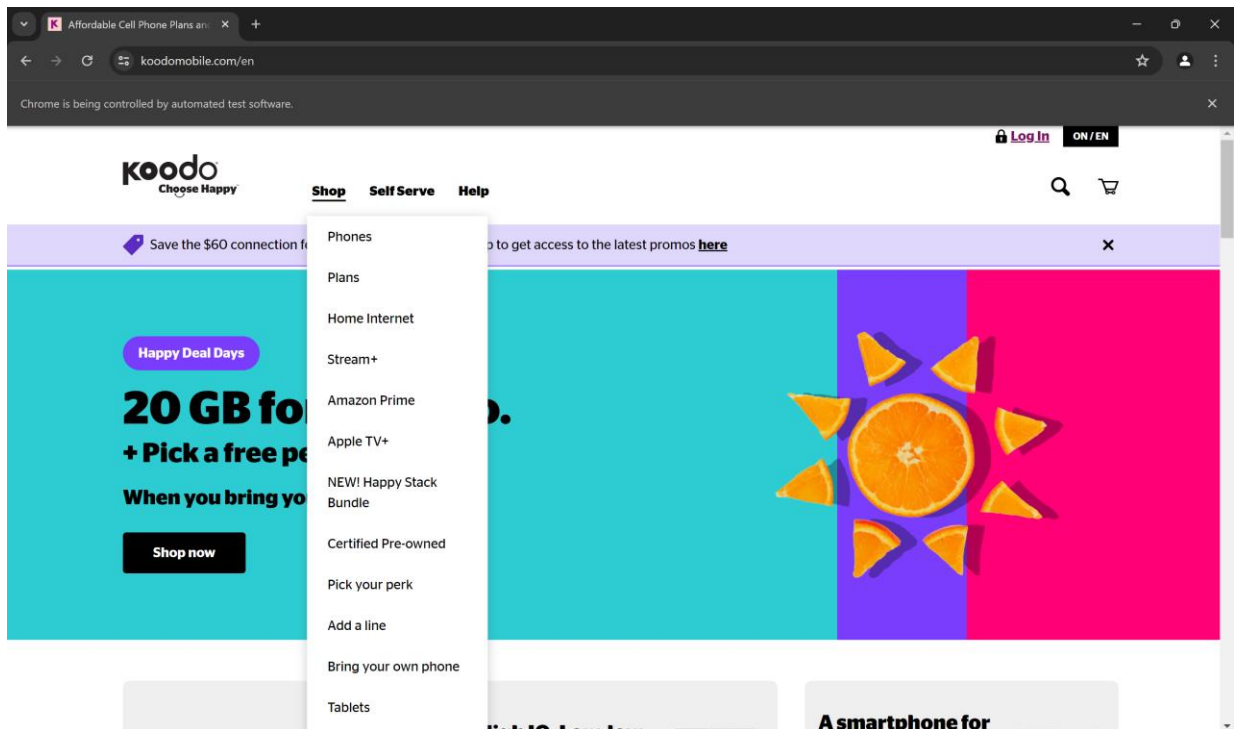


Figure 10 : Homepage of Koodo Mobile
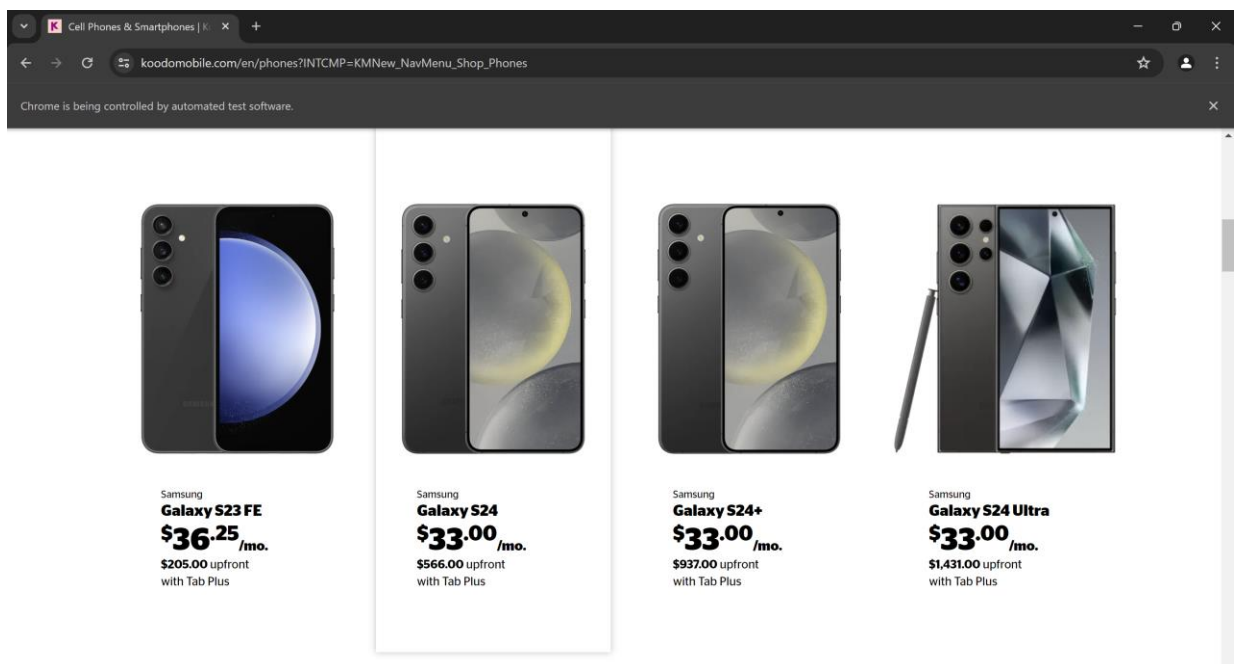
Figure 11 : Dropdown Button



Figure 12 : Phones Page

Figure 13 : Plans Page