Code Explanation

This following script/code will use Selenium to log in to Amazon, scrape the Best sellers section for specific product details in selected categories, and save the collected data to a CSV file.

1. Import Statements

`time`: Used for adding delays to ensure pages load fully.

`pandas`: For saving the scraped data into a structured CSV file.

`selenium`: For web automation and scraping.

- `webdriver`: Controls the browser.
- `By`: Locates HTML elements.
- `expected_conditions (ec)`: Helps wait for specific conditions (e.g., element presence).
- `WebDriverWait`: Adds explicit waits to avoid timing issues.
- `TimeoutException, NoSuchElementException`: Handles common Selenium exceptions.

2. Constants

- `AMAZON_LOGIN_URL`: URL for the Amazon login page.
- `BEST_SELLERS_URL`: Base URL for the Amazon Best Sellers page.
- `CATEGORIES`: A list of categories to scrape. Each corresponds to a subdirectory of the Best Sellers page.
- `OUTPUT_FILE`: Filename for saving the scraped data.

3. Selenium WebDriver Setup

- `driver = webdriver.Chrome()`: Launches Chrome WebDriver.
- `wait = WebDriverWait(driver, 10)`: Configures explicit waits of up to 10 seconds for elements.

4. `login_amazon()` Function

- Automates Amazon login using provided email and password.
- Steps:
    1. Navigate to the login page (`AMAZON_LOGIN_URL`).
    2. Wait for the email field (`ap_email`) to load, input the email, and click "Continue."

3. Wait for the password field (`ap_password`), input the password, and click "Sign In."
4. If successful, print "Login successful." Otherwise, handle timeouts and quit the browser.

5. `scrape_category()` Function

Scrapes product details for a given category URL.

Steps:

1. Navigate to the category URL.
2. Locate products on the page (`div.zg-grid-general-face-out`).
3. Extract details for each product:
    - **`name`**: Product name.
    - **`price`**: Product price.
    - **`discount`**: Discount information.
    - **`rating`**: Customer rating.
    - **`sold_by`**: Seller information.
    - **`ship_from`**: (Placeholder for this example).
    - **`images`**: List of product image URLs.
4. Handle pagination to scrape multiple pages (up to 30 or 1500 products).
5. Return a list of dictionaries, each representing a product.

6. `save_data()` Function

- Converts the scraped data into a Pandas DataFrame.
- Saves the data to a CSV file (`OUTPUT_FILE`).
- Prints a confirmation message.

7. `Main` Block

Ensures the script runs only when executed directly.

Steps:

1. Define Amazon credentials (`EMAIL` and `PASSWORD`).
2. Log in using `login_amazon()`.
3. Loop through the `CATEGORIES`:
    - Build the category URL.
    - Scrape the category using `scrape_category()`.
    - Append the results to `all_data`.
4. Save all the scraped data to a CSV file using `save_data()`.

5. Quit the browser in all cases using the `finally` block.

Example Flow

1. **Log in**: User credentials are entered, and the login is automated.
2. **Scrape Categories**:
    - For each category (e.g., `kitchen`, `shoes`), products are scraped.
    - Data includes product name, price, discount, images, and rating.
    - Pagination ensures data is collected across multiple pages.
3. **Save Data**:
    - All scraped product data is combined.
    - The results are saved in a CSV file (`amazon_best_sellers.csv`).

Error Handling

1. **Login Failure**: Timeout exceptions are caught, and the browser quits.
2. **Scraping Issues**: Missing elements or navigation errors are handled using *`NoSuchElementException`*.
3. **General Errors**: Any unexpected exceptions during scraping are logged, and the script proceeds or terminates gracefully.

Output

The output file (`amazon_best_sellers.csv`) contains structured product data for all specified categories, including details like product name, price, discount, rating, seller, and image urls.

CODE:

```
import time

import pandas as pd
from selenium import webdriver
from selenium.common.exceptions import TimeoutException, NoSuchElementException
from selenium.webdriver.common.by import By
from selenium.webdriver.support import expected_conditions as ec
from selenium.webdriver.support.ui import WebDriverWait

AMAZON_LOGIN_URL = "https://www.amazon.in/ap/signin"
BEST_SELLERS_URL = "https://www.amazon.in/gp/bestsellers/"
CATEGORIES = [
   "kitchen", "shoes", "computers", "electronics"
]
OUTPUT_FILE = "amazon_best_sellers.csv"

driver = webdriver.Chrome()
```

```
wait = WebDriverWait(driver, 10)


def login_amazon(email, password):
    """Logs into Amazon using the provided credentials."""
    driver.get(AMAZON_LOGIN_URL)
    try:
        email_input = wait.until(ec.presence_of_element_located((By.ID,
"ap_email")))
        email_input.send_keys(email)
        driver.find_element(By.ID, "continue").click()

        password_input = wait.until(ec.presence_of_element_located((By.ID,
"ap_password")))
        password_input.send_keys(password)
        driver.find_element(By.ID, "signInSubmit").click()
        print("Login successful")
    except TimeoutException:
        print("Login failed. Check credentials or page load time.")
        driver.quit()


def scrape_category(category_urls):
    """Scrapes data for a given category."""
    driver.get(category_urls)
    time.sleep(2)

    products = []
    try:
        for _ in range(30):
            product_elements = driver.find_elements(By.CSS_SELECTOR,
"div.zg-grid-general-face-out")
            for product in product_elements:
                try:
                    name = product.find_element(By.CSS_SELECTOR,
".p13n-sc-truncate-desktop-type2").text
                    price = product.find_element(By.CSS_SELECTOR,
".p13n-sc-price").text
                    discount = product.find_element(By.CSS_SELECTOR,
".a-row.a-size-small").text
                    rating = product.find_element(By.CSS_SELECTOR,
".a-icon-alt").text
                    sold_by = product.find_element(By.CSS_SELECTOR,
".a-size-small.a-color-secondary").text
                    ship_from = "Amazon"
                    images = product.find_elements(By.CSS_SELECTOR, ".zg-item
img")

                    product_data = {
```

```
                        "Product Name": name,
                        "Price": price,
                        "Discount": discount,
                        "Rating": rating,
                        "Sold By": sold_by,
                        "Ship From": ship_from,
                        "Images": [img.get_attribute("src") for img in images],
                        "Category": category_urls.split("/")[-1]
                    }
                    products.append(product_data)
                except NoSuchElementException:
                    continue

            try:
                next_button = driver.find_element(By.CSS_SELECTOR, "li.a-last
a")
                next_button.click()
                time.sleep(2)
            except NoSuchElementException:
                break
    except Exception as e:
        print(f"Error scraping category: {e}")

    return products


def save_data(data, file_name):
    """Saves the scraped data to a CSV file."""
    df = pd.DataFrame(data)
    df.to_csv(file_name, index=False)
    print(f"Data saved to {file_name}")


if __name__ == "__main__":
    try:

        EMAIL = "your_email@example.com"
        PASSWORD = "your_password"

        login_amazon(EMAIL, PASSWORD)

        all_data = []
        for category in CATEGORIES:
            print(f"Scraping category: {category}")
            category_url =
f"{BEST_SELLERS_URL}{category}/ref=zg_bs_nav_{category}_0"
            category_data = scrape_category(category_url)
            all_data.extend(category_data)
```

```
        save_data(all_data, OUTPUT_FILE)
    finally:
        driver.quit()
```