Security Project

2025

WEB SCRAPING

TAYARA.TN

TEAM

Eslem Sebri

Oumayma Abayed

Overview of the Solution

This document presents the design, system components, and data flow of our Tayara.tn vehicle listing scraper, extended with data encryption and decryption. The goal is to securely collect, store, and protect web-scraped vehicle data.

tayara

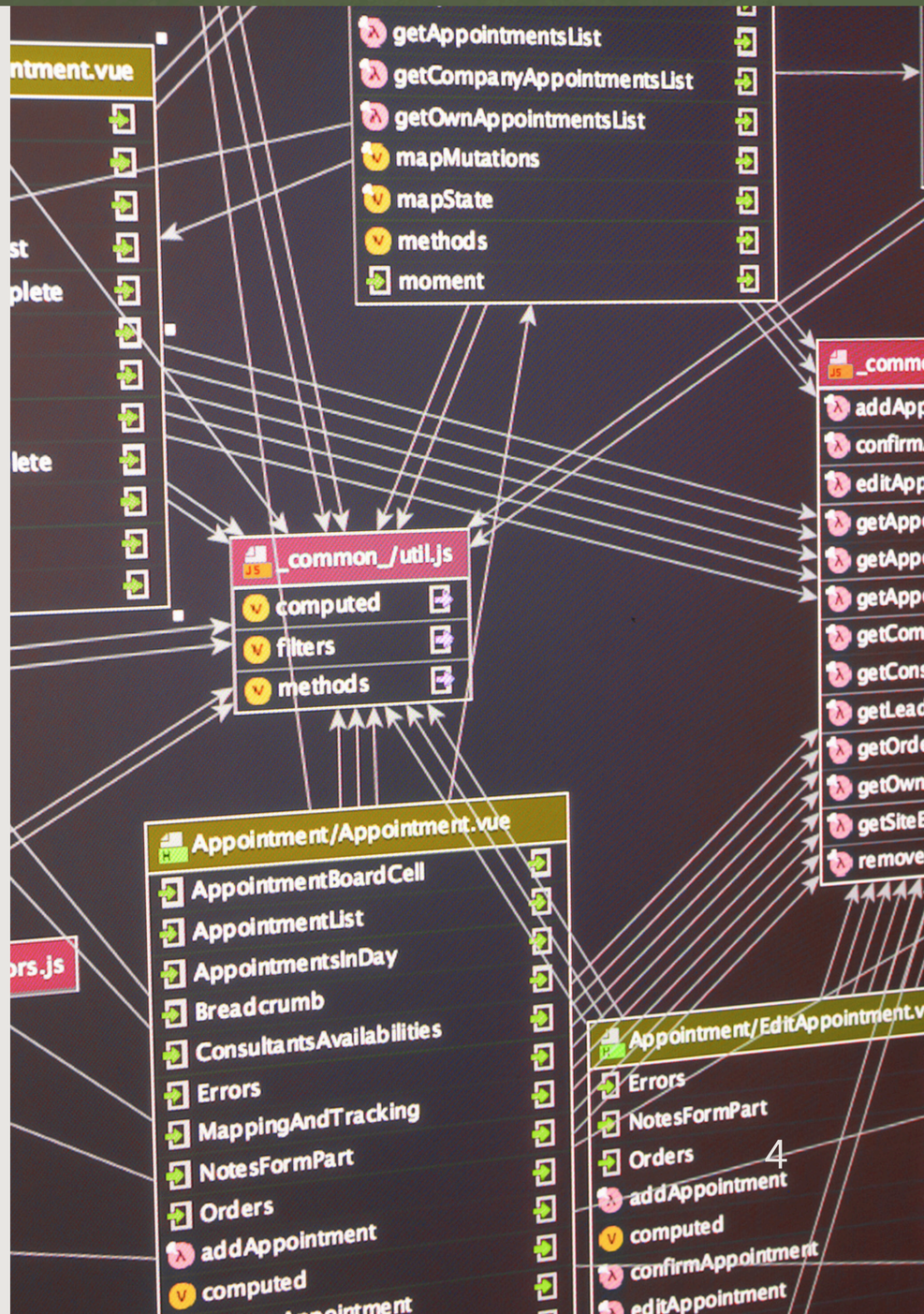
System Components & Roles

Component	Role/Function
Selenium Web Scraper	Automates interaction with Tayara.tn pages and extracts listing information
Parser (In-Loop)	Extracts title, price, and location from HTML elements
Storage (CSV Writer)	Uses Pandas to store structured data into a CSV file
Encryptor	Encrypts the resulting CSV using Fernet symmetric encryption
Decryptor	Decrypts the encrypted CSV when needed for access or processing

Workflow & Data Flow

Step-by-Step Flow:

1. **Initialize WebDriver** with headless Chrome options.
2. **Navigate Tayara Pages** from 1 to 20 using a for loop.
3. **Extract Listings** from each page:
    a. Extract <h2> tag (title)
    b. Extract <data> tag (price)
    c. Extract location from .text-neutral-500
    d. Build and store full link to each listing
4. **Append Data** to a Python list.
5. **Write to CSV** using pandas.DataFrame.to_csv.

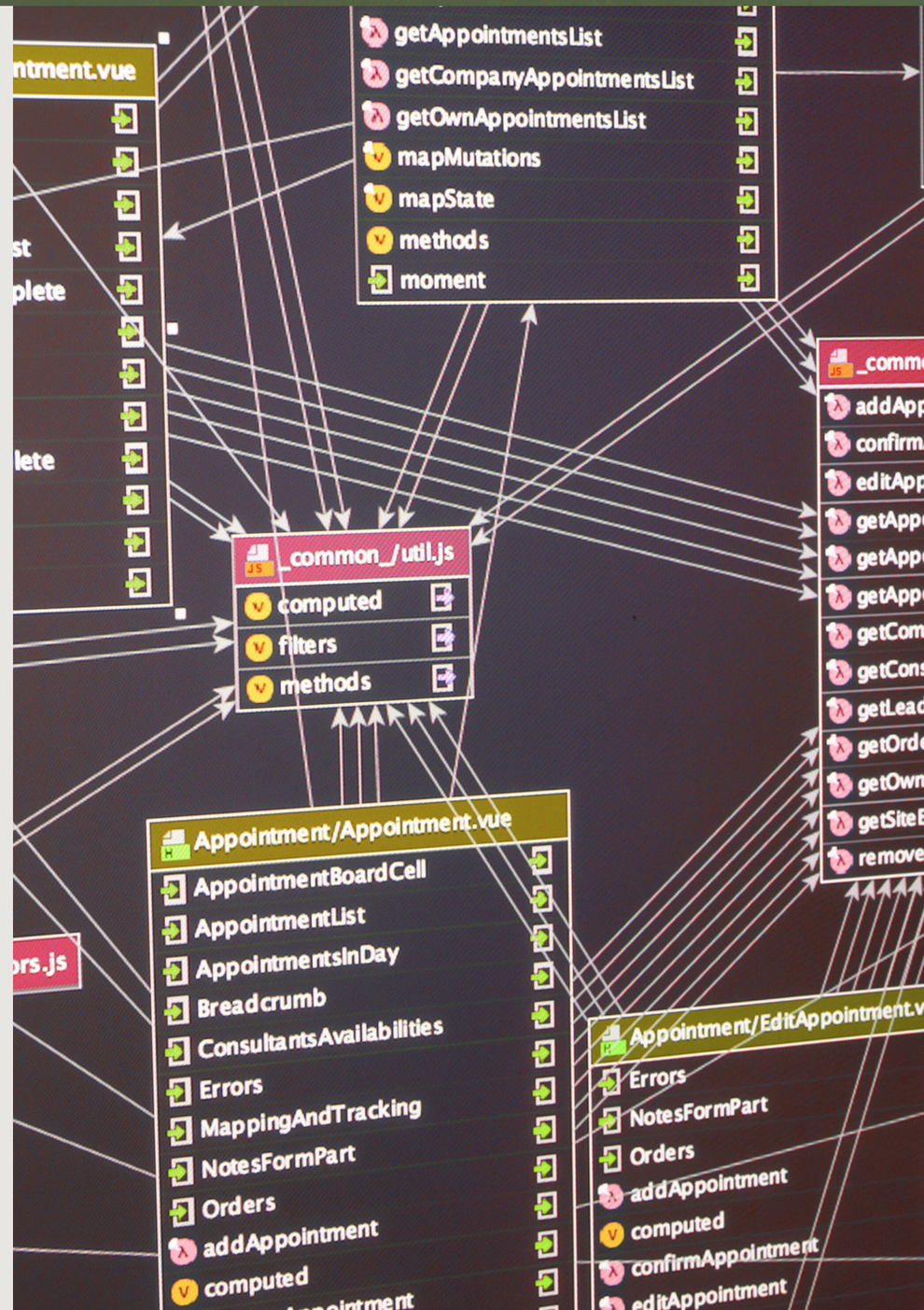


6. Encryption:

- Generate or load Fernet key
- Encrypt tayara_vehicles.csv content
- Save encrypted file as tayara_vehicles_encrypted.csv
- Save key as encryption.key

7. Decryption:

- Load key from encryption.key
- Read and decrypt tayara_vehicles_encrypted.csv
- Save result as tayara_vehicles_decrypted.csv



Exchanged Data

From	To	Data	Format
Selenium	Python Parser	HTML Element Data	DOM / Text
Parser	Data Store	Structured listing info	Python dict
Data Store	Disk	Final dataset	CSV file
Encryptor	Disk	Encrypted file	CSV file
Decryptor	CSV	Decrypted CSV data	CSV file

Tools & Technologies

1

Python

Programming language

2

Selenium

Web automation & scraping

3

ChromeDriver

Headless browser automation

4

Pandas

Data manipulation & storage

5

CSV

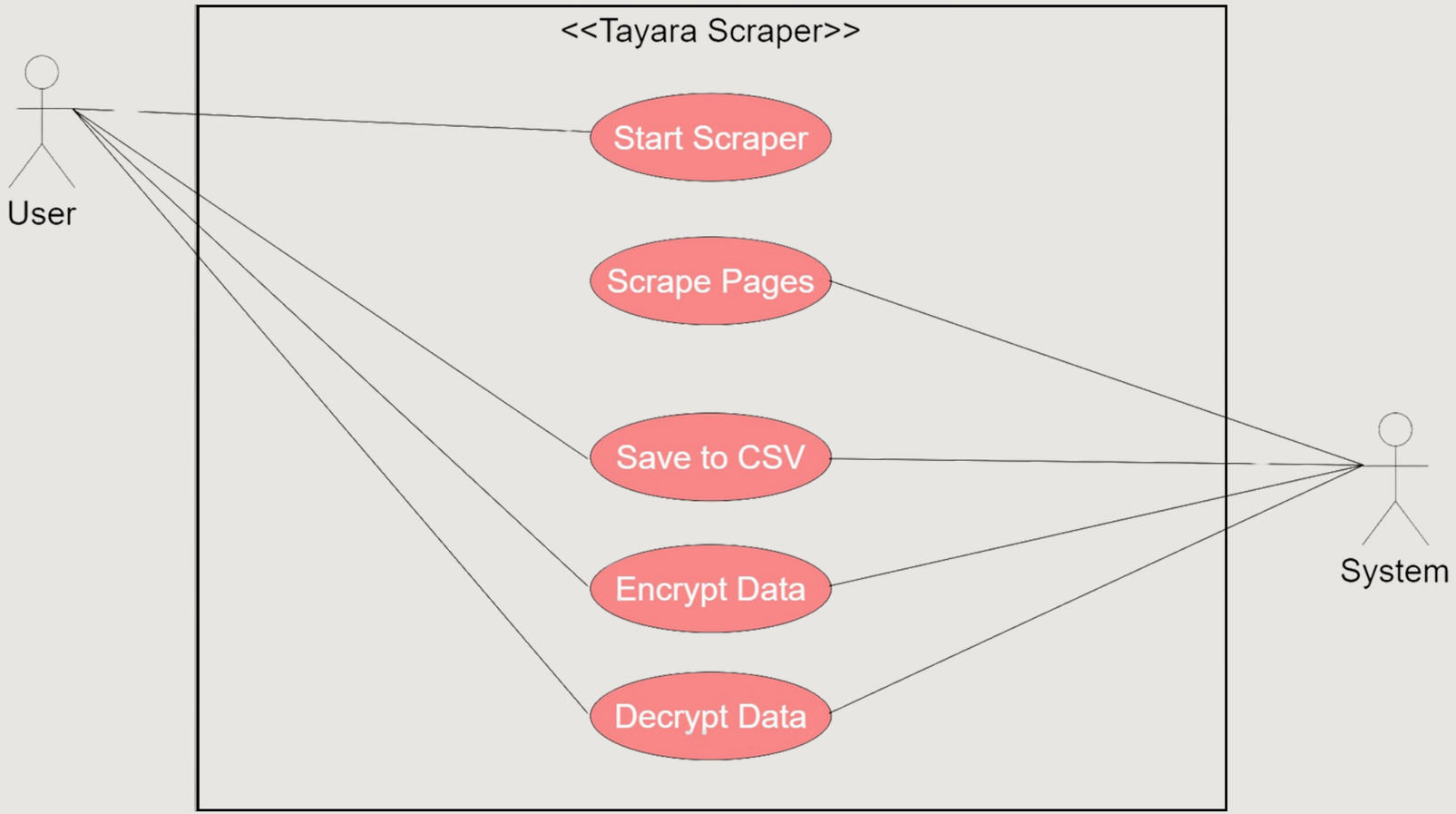
Flat-file data format

6

Cryptography

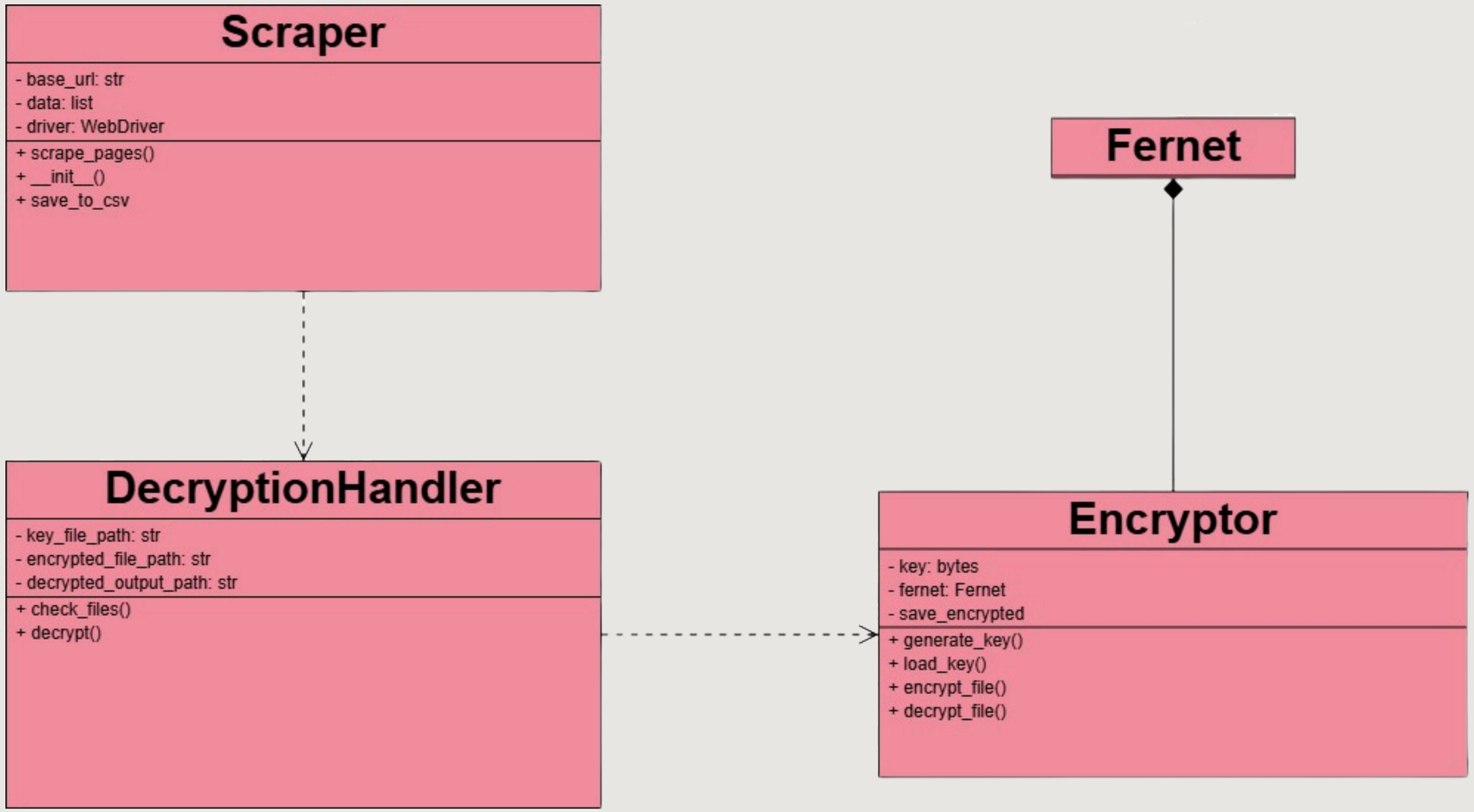
Fernet-based AES encryption/decryption

Actors & Use Cases Diagram



- **User (Scraper Operator):** Initiates and manages the overall process of scraping, encryption, decryption, and saving data.
- **System:** Executes the scraping, encryption, decryption, and file handling processes in the background

Class Diagram



Development Phases

Scraping

Phase 1: Environment Setup

Install Python, Selenium, ChromeDriver, and Pandas

Phase 2: Build Initial Scraper

Create functional script to scrape one page

Phase 3: Expand to Pagination

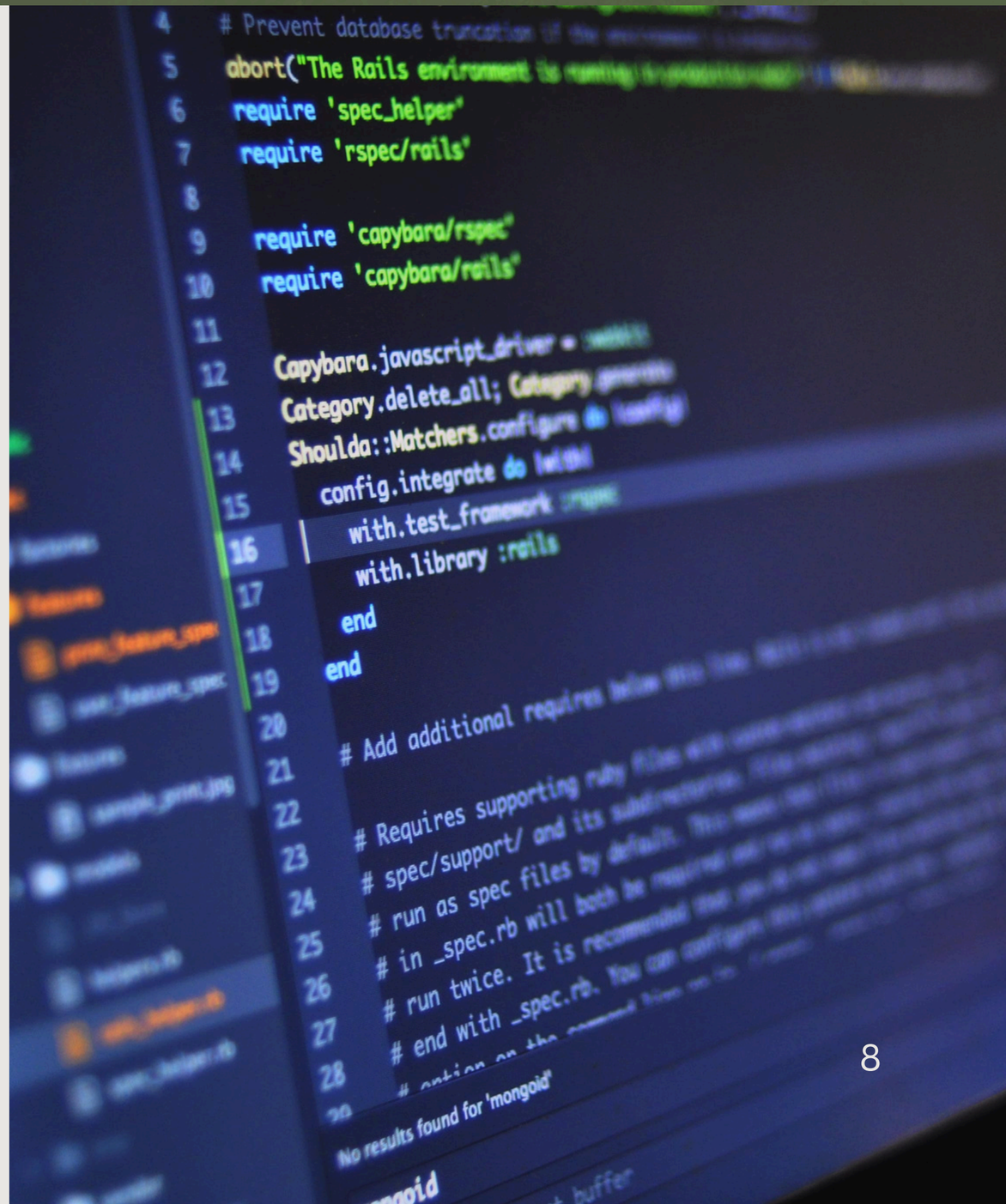
Loop through multiple pages (20 pages)

Phase 4: Error Handling

Add try/except to skip malformed listings

Phase 5: CSV Export

Store clean structured data



Encryption

Development Phases

Phase 6: Key generation

Generated and saved a Fernet encryption key.

Phase 7: CSV Encryption

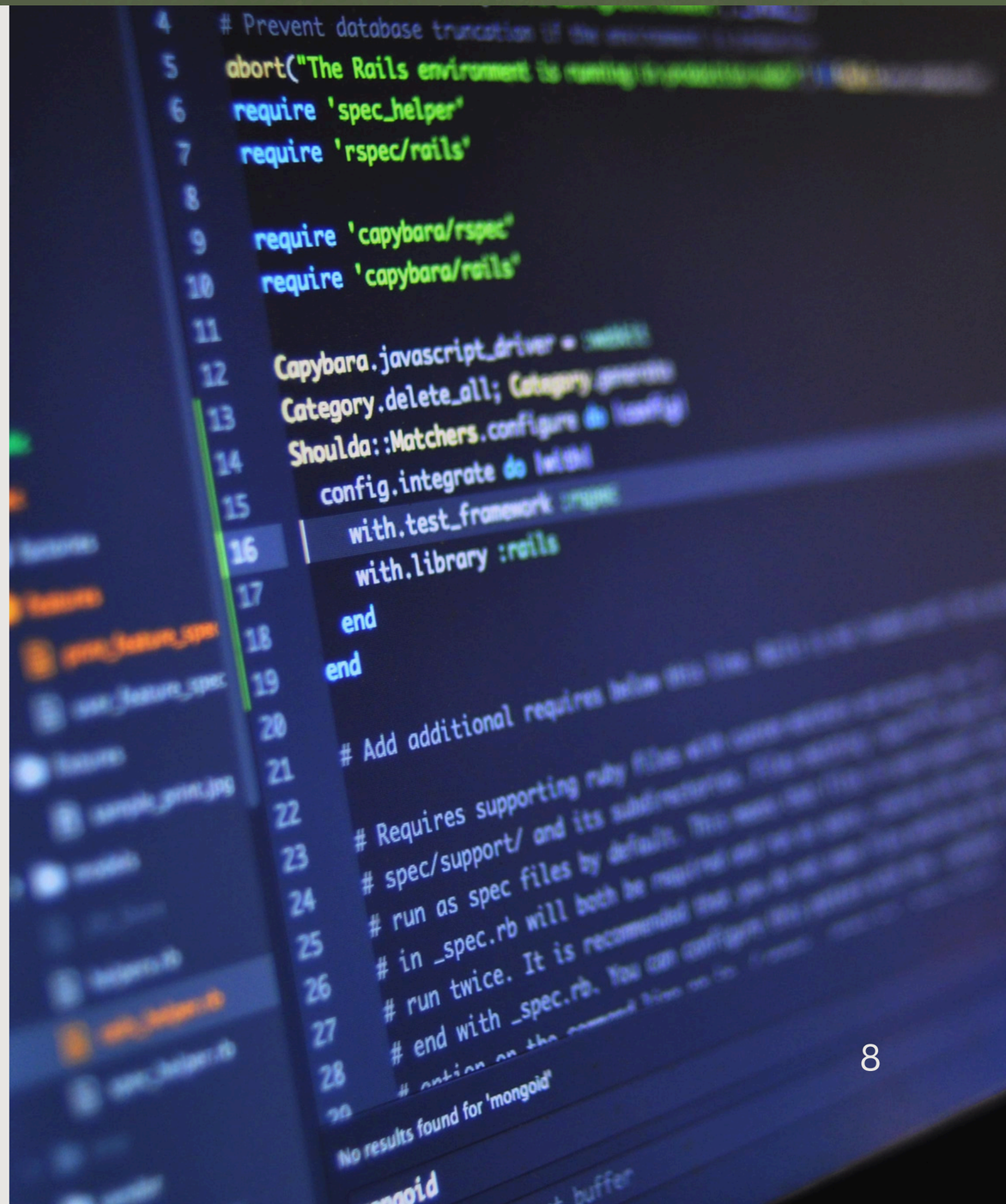
Encrypted the CSV file using the saved key.

Phase 8: Decryption Logic

Decrypted the file using the same Fernet key.

Phase 9: File Handling Fixes

Resolved filename/path errors for smooth execution.



Python Code

Web Scraping

scraper.py 1

C: > Users > user > Downloads > scraper.py > ...

```python
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.options import Options
import pandas as pd
import time

# Setup Chrome with headless option (remove headless if you want to see it run)
options = Options()
options.add_argument('--headless')
options.add_argument('--no-sandbox')
options.add_argument('--disable-dev-shm-usage')

# Path to ChromeDriver (you can modify if needed)
service = Service()

driver = webdriver.Chrome(service=service, options=options)

# URL base
base_url = 'https://www.tayara.tn/ads/c/V%C3%A9hicules/?page='

# Data container
data = []
```

```python
# Loop over 20 pages
for page in range(1, 21):
    driver.get(base_url + str(page))
    time.sleep(2)  # Let the page load

    # Get all listings
    listings = driver.find_elements(By.CSS_SELECTOR, 'a[href^="/item/"]')

    for listing in listings:
        try:
            title = listing.find_element(By.TAG_NAME, 'h2').text
            price = listing.find_element(By.TAG_NAME, 'data').text

            # Get last span with neutral-500 (contains location and time)
            location_time = listing.find_elements(By.CSS_SELECTOR, 'span.text-neutral-500')[-1].text
            location = location_time.split(',')[0].strip()

            # Build full link
            partial_link = listing.get_attribute('href')

            data.append({
                'Title': title,
                'Price': price,
                'Location': location,
                'Link': partial_link
            })
        except Exception as e:
            # Skip any listings that are malformed
            continue

    print(f'✅ Page {page} scraped')

# Close browser
driver.quit()

# Save to CSV
df = pd.DataFrame(data)
df.to_csv('tayara_vehicles.csv', index=False, encoding='utf-8-sig')
print("✅ Scraping completed and data saved to 'tayara_vehicles.csv'")
```

Python Code

Encryption

encrypter.py > ...

```python
from cryptography.fernet import Fernet

# Generate a key (only run this ONCE and save the key)
key = Fernet.generate_key()

# Save the key securely
with open("encryption.key", "wb") as key_file:
    key_file.write(key)

# Initialize Fernet with the key
fernet = Fernet(key)

# Read the CSV file to encrypt
with open("tayara_vehicles.csv", "rb") as file:
    original = file.read()

# Encrypt the data
encrypted = fernet.encrypt(original)

# Save the encrypted data to a file
with open("tayara_vehicles_encrypted.csv", "wb") as enc_file:
    enc_file.write(encrypted)

print("✅ tayara_vehicles.csv has been encrypted.")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
✅ tayara_vehicles.csv has been encrypted.
PS C:\Users\MSI\Desktop\JUNIOR\SECURITY\Projet> & C:/Users/MSI/AppData/Local/Programs/Python/Python313/python.exe
rypter.py
```

Python Code

Decryption

Welcome | scraper.py | encrypter.py | decrypter.py

decrypter.py > ...

```python
# Initialize Fernet
fernet = Fernet(key)

# Read the encrypted file
with open(encrypted_file_path, "rb") as enc_file:
    encrypted_data = enc_file.read()

# Decrypt the data
try:
    decrypted_data = fernet.decrypt(encrypted_data)
except Exception as e:
    print(f"❌ Decryption failed: {e}")
    exit()

# Save the decrypted data
with open(decrypted_output_path, "wb") as dec_file:
    dec_file.write(decrypted_data)

print(f"✅ File decrypted successfully and saved as '{decrypted_output_path}'.")
```

Welcome | scraper.py | encrypter.py | decrypter.py

decrypter.py > ...

```python
from cryptography.fernet import Fernet
import os

# File paths
key_file_path = "encryption.key"
encrypted_file_path = "tayara_vehicles_encrypted.csv"
decrypted_output_path = "tayara_vehicles_decrypted.csv"

# Check if necessary files exist
if not os.path.exists(key_file_path):
    print(f"❌ Encryption key file '{key_file_path}' not found.")
    exit()

if not os.path.exists(encrypted_file_path):
    print(f"❌ Encrypted file '{encrypted_file_path}' not found.")
    exit()

# Load the encryption key
with open(key_file_path, "rb") as key_file:
    key = key_file.read()
```

PROBLEMS | OUTPUT | DEBUG CONSOLE | TERMINAL | PORTS

```
rypter.py
✅ File decrypted successfully and saved as 'tayara_vehicles_decrypted.csv'.
PS C:\Users\MSI\Desktop\JUNIOR\SECURITY\Projet> & C:/Users/MSI/AppData/Local/
rypter.py
```

THANK YOU

For Following Along