KYC Web Scraping Project

1. Project Title

Alaska Legislature Member Data Extraction using Java, Playwright, and Spring Boot

2. Objective

The main objective of this project is to **automatically extract structured information** about **Alaska State Senators** from the official legislature website
🔗 https://akleg.gov/senate.php
and store the details in a **clean JSON format** for future analysis or integration with APIs or dashboards.

3. Tools and Technologies Used

Category	Tools / Technologies
Programming Language	**Java 21**
Framework	**Spring Boot**
Automation Library	**Playwright (Java)**
Build Tool	**Gradle**
JSON Handling	**Gson Library**
IDE Used	**IntelliJ IDEA**
Output Format	**JSON File**

4. Project Overview

This project automates the process of collecting information from the **Alaska State Legislature** website.
It uses **Playwright** to control a headless Chromium browser and navigates through each senator's profile page to extract:

- Full Name
- Title (Senator)
- District / Position
- Party Affiliation
- Session and Interim Contact Addresses
- Phone Numbers
- Email Address
- Official Profile URL

The extracted data is serialized into a single structured JSON file named:
`ak_senate_members.json`

5. Step-by-Step Workflow

Step 1: Setup and Initialization

- Configured a **Spring Boot project** using Gradle build system.
- Installed **Playwright** for Java (`gradlew playwrightInstall`).
- Created the necessary project structure (controllers, services, utils, models).

Step 2: Website Analysis

- Inspected the structure of the Alaska Senate webpage using Chrome DevTools.
- Located relevant HTML tags for name, email, phone, and address details.

- Identified consistent patterns for senator detail pages (`/basis/Member/Detail/`).

Step 3: Data Extraction

- Used Playwright to:
    - Open the main Senate page.
    - Collect all senator profile URLs.
    - Visit each senator’s page individually.
    - Extract text content based on class names and tag structure.

Step 4: Data Cleaning

- Removed unwanted newlines and whitespace from senator names.
- Merged multi-line addresses into readable single strings.
- Replaced missing or confidential addresses with placeholders when necessary.

Step 5: Data Serialization

- Created a **POJO class (Member.java)** to map all senator data fields.
- Stored all entries in a List and converted it to JSON using **Gson**.
- The output JSON file was generated automatically after execution.

Step 6: Testing and Verification

- Verified extracted JSON by cross-checking random senator pages manually.
- Ensured that all fields (email, address, name, etc.) were correctly mapped.
- Adjusted locators for consistency and reliability across all profiles.

6. Output Example

```
{
  "name": "Matt Claman",
  "title": "Senator",
  "position": "District H",
  "party": "Democrat",
  "address": "State Capitol Room 429, Juneau, AK 99801 | 1500 W Benson Blvd, Anchorage, AK
99503",
  "phone": "907-465-4919",
  "email": "Senator.Matt.Claman@akleg.gov",
  "url": "https://www.akleg.gov/basis/Member/Detail/34?code=cla"
}
```

All data is formatted and saved in a file named `ak_senate_members.json`.

7. Learning Outcomes / Notes

- Learned how to use **Playwright with Java** for automating web data extraction.
- Understood **DOM traversal**, **selectors**, and **data cleaning techniques** for structured scraping.
- Implemented **Spring Boot service architecture** to modularize the scraper logic.
- Used **Gradle** for dependency management and build automation.
- Gained experience in **data serialization using Gson**.
- Enhanced debugging and error-handling skills while dealing with dynamic HTML and inconsistent data fields.

8. Challenges Faced

- Some senator profiles contained **hidden or confidential addresses**, requiring conditional handling.

- Extracting names from formatted headings (with multiple lines and years) required string cleaning.
- Playwright occasionally timed out on slower connections — handled using retries.
- Maintaining data consistency across all 20 senators needed precise locator matching.

9. Time Taken to Complete the Project

Task	Description	Time Taken
Project Setup	Created Spring Boot + Gradle project, installed Playwright	45 minutes
Website Structure Analysis	Studied DOM layout, located tags for extraction	30 minutes
Scraper Logic Implementation	Developed Playwright scraping script	1 hour 30 minutes
Data Cleaning & JSON Generation	Cleaned and structured extracted data	1 hour
Testing & Verification	Checked extracted results for correctness	45 minutes
Documentation & README Preparation	Wrote explanation, structure, and notes	30 minutes
Total Estimated Time	—	**≈ 5 hours**

10. Conclusion

This project successfully demonstrates **automated web data extraction using Playwright and Java** integrated within a Spring Boot framework.
It highlights skills in **web automation, data processing, and JSON handling** while maintaining modular, maintainable code.

The scraper efficiently gathers real-world data in a structured format and can easily be adapted for similar government or institutional websites.

11. Submitted By

Name: Krishna Gupta
Enrollment No.: 11214803122
Branch: Information Technology
Institute: Maharaja Agrasen Institute of Technology, Delhi
Email: krishnagupta2380@gmail.com
Date of Submission: November 4, 2025