Task: **Build a Simple Device Matching Backend Service**

Objective:

Create a backend service for a simple device matching service using Java and Spring Boot. The service should have an ednpoint which accepts User-Agent strings and is able to match a Device with the same characteristics (OS name, OS version, Browser name, Browser version). The service should use Aerospike as the database (Community version can be found here: https://aerospike.com/download/?software=server-community).

Requirements:

Project Setup:

- Use **Java 11** or higher.
- Use **Spring Boot** (latest version) for application setup.
- Integrate **Aerospike** as the database.
- Use **Maven** or **Gradle** as the build tool.
- Create unit tests.
- Create integration/component tests (optional).

Functionality: Implement the following features:

Device Matching:

- **Match Device:** Endpoint to create a new device profile or match an existing one. The endpoint should accept a User-Agent string and the service should be able to extract OS name, OS version, Browser name and Browser version from it. Device information stored in the database should contain the following information:
    - **Device ID –** a unique device identifier
    - **HitCount –** number of times this device has been seen
    - **OS name –** Operating system name (eg. MacOS, Windows, etc.)
    - **OS version –** Operating system version
    - **Browser name –** Browser name (eg. Chrome, Firefox, etc.)
    - **Browser version –** Browser version

When a new request comes with the same identifiers (OS name, OS version, Browser name and Browser version) for an existing device, the service should be able to match to the existing device profile and the hit count should be incremented.

Device Management:

- **Get a Device by ID:** Endpoint to retrieve details for a device by its ID.
- **Get all Devices for a given OS name:** Endpoint to retrieve a list of all devices which have a common operating system.
- **Delete a Device/s:** Endpoint to delete a device profile by ID(s).

Testing:

- **Unit Tests:** Write unit tests for the service layer using JUnit and Mockito.
- **Component Tests:** Write component tests for the REST controllers, focusing on end-to-end testing of the API (optional).

Documentation:

- Include a brief README file explaining how to set up, run, and test the application.

Deliverables:

- Source code in a GitHub repository (or a zip file).
- Instructions for running the application, including any prerequisites.

Evaluation Criteria:

- Code quality and organization.
- Adherence to the specified requirements.
- Proper use of Spring Boot and Aerospike.
- Test coverage and test quality.
- API design and RESTful principles.
- Documentation and clarity of setup instructions.