**Backend API Development Test - Java Spring Boot**

**Time Limit:** 90 Minutes

Objective:

Create a RESTful API using Java (version 17 or higher) and the Spring Boot framework. This API will manage user data, mimicking a typical backend service.

**Requirements:**

1. **Java & Spring Boot:**
   * The API must be written in Java (version 17+).
   * Utilize the Spring Boot framework.
2. **RESTful Endpoints:** Implement the following RESTful API endpoints for user management:
   * GET /users: Retrieve a list of all users.
   * GET /users/{userId}: Retrieve details of a specific user. Return an appropriate HTTP status code (e.g., 404 Not Found) if the user does not exist.
   * POST /users: Create a new user. Return an appropriate HTTP status code (e.g. 201 Created)
   * PUT /users/{userId}: Update details of an existing user. Return an appropriate HTTP status code (e.g., 404 Not Found) if the user does not exist.
   * DELETE /users/{userId}: Delete a specific user. Return an appropriate HTTP status code (e.g., 404 Not Found) if the user does not exist.
3. **Data Model (Simplified):**
   * Use the data structure from https://jsonplaceholder.typicode.com/users as a *reference*.
   * For this test, focus on the following fields for the User model to keep it manageable within the time limit:
     + id (Long)
     + name (String)
     + username (String)
     + email (String)
     + phone (String)
     + website (String)
   * **Data Storage:** For simplicity within the 90-minute timeframe, store user data in memory (e.g., a List or Map within a service class). **Do not worry about connecting to a database for this test.**
   * Initialize the in-memory store with a few sample users (you can copy a few entries from the jsonplaceholder URL).
4. **Validation:**
   * For POST and PUT requests, ensure that name, username, and email fields are provided (i.e., not null or empty).
   * Return an appropriate HTTP status code (e.g., 400 Bad Request) and a meaningful error message if validation fails.
5. **Response Format & HTTP Status Codes:**
   * All request and response bodies must be in JSON format.
   * Include appropriate HTTP status codes for all success (e.g., 200 OK, 201 Created, 204 No Content) and error (e.g., 400 Bad Request, 404 Not Found) scenarios.

**Optional (Bonus Points):**

These are considered bonus points and not strictly required to complete the main task within the 90-minute timeframe. Candidates are encouraged to attempt these if they finish early.

* **Unit Tests:** Add unit tests for at least one or two API endpoints (e.g., GET /users, POST /users) to demonstrate testing principles. Focus on testing the service layer or controller logic with mocks.
* **Dockerfile:** Provide a basic Dockerfile to containerize the Spring Boot application.
* **CI/CD Script:** Briefly outline or provide a very basic snippet of a CI/CD script (e.g., for GitHub Actions, Jenkinsfile) that would build and potentially deploy this service.