# **CRUD Service API**

This document provides instructions and details for setting up, understanding, and running the CRUD Service API on a Windows 11 environment.

## **Table of Contents**

* Overview
* Setup Instructions
  + Prerequisites
  + Installation
* Code Architecture
* Endpoints
* Performance Metrics
* Retry Mechanism

## **Overview**

The CRUD Service API allows users to create, read, update, and delete data. The service utilizes:

* **Spring Boot** as the framework
* **Redis** for caching
* **Micrometer** for performance metrics
* **Spring Retry** for retry logic

## **Setup Instructions**

### **Prerequisites**

1. **Java Development Kit (JDK):** Ensure JDK 17 or later is installed.
   * [Download JDK](https://www.oracle.com/java/technologies/javase-jdk11-downloads.html)
2. **Maven:** Install Maven for dependency management.
   * [Download Maven](https://maven.apache.org/download.cgi)
3. **Redis:** Install Redis for caching.
   * Use Windows Subsystem for Linux (WSL) to run Redis if needed.
   * [Guide to install Redis on Windows](https://redis.io/docs/getting-started/installation/)
4. **Git:** Ensure Git is installed for cloning the repository.
   * [Download Git](https://git-scm.com/)

### **Installation**

1. Clone the repository:  
   git clone https://github.com/your-repo/crud-service-api.git
2. Navigate to the project directory:  
   cd crud-service-api
3. Build the project using Maven:  
   mvn clean install
4. Run the application:  
   mvn spring-boot:run

## **Code Architecture**

### **Controller Layer**

* **CrudController:** Handles HTTP requests for CRUD operations and maps them to service layer methods.

### **Service Layer**

* **DataProcessor:** Implements business logic for processing, caching, and managing retry mechanisms.

### **Repository Layer**

* **DataRepository:** Extends JpaRepository for database interactions.

### **Utilities**

* **Micrometer:** Captures performance metrics like request timing.
* **Spring Retry:** Ensures resilience by retrying failed operations.

## 

## 

## 

## 

## **Endpoints**

### **Create Data**

* **POST /api/data**

Request Body:  
{

"key1": "yourKey",

"value1": "yourValue"

}

* Response: Data created successfully.

### **Read Data**

* **GET /api/data?key={key}**
* Response: Data value or Data not found.

### **Update Data**

* **PUT /api/data?key={key}&value={value}**
* Response: Data updated successfully.

### **Delete Data**

* **DELETE /api/data?key={key}**
* Response: Data deleted successfully.

## **Performance Metrics**

* **Timer Metrics:**
  + data.process.time
  + data.update.time
  + data.delete.time
* These metrics can be integrated with monitoring tools like Prometheus and Grafana.

## **Retry Mechanism**

* **Update and Delete:**
  + Configured to retry up to 3 times with a delay of 2000ms between attempts.
  + Annotation: @Retryable

## **Notes**

* Ensure Redis is running before starting the application.
* Use tools like Postman to test the endpoints.
* For additional configurations, refer to the application.properties file.