To create project, structured approach to build a Todo application using Spring Boot, implementing RESTful endpoints for CRUD operations.

Below are the steps and instructions through the process:

**1. Project Setup : (Code added in github)**

First, set up Spring Boot project using either Spring Initializr method. including the necessary dependencies in pom.xml (if using Maven):

* spring-boot-starter-web: For building RESTful APIs.
* spring-boot-starter-data-jpa: For data access using Spring Data JPA.
* spring-boot-starter-validation: For validation of request parameters and bodies.
* spring-boot-starter-logging: For logging using SLF4J and Logback.
* spring-boot-starter-actuator: For monitoring and managing the application.
* springdoc-openapi-ui: For generating API documentation using Swagger.

**2. Create JavaProjectApplication Class : (Code added in github)**

Created main package name as com.NPCI inside src/main/java in project

Package com.NPCI

Class : JavaProjectApplication.java

**3. Create Entity Class : (Code added in github)**

Created entiry package inside src/main/java in project:

package com.NPCI.entity;

Class : Todo.java

**4. Create Repository Interface : (Code added in github)**

package com.NPCI.repo;

Class : TodoJPARepository.java

interface TodoSpringDataJPARepository extends JpaRepository

**5. Create Service Layer : (Code added in github)**

Implement a TodoService.java class to encapsulate business logic:

package com.NPCI.service;

TodoService implements TodoServiceContract

interface TodoServiceContract

**6. Create Controller Class : (Code added in github)**

Implement the TodoController class to define RESTful endpoints:

package com.NPCI.service;

class : TodoController.java

**7. Created application.properties in src/main/resources : (Code added in github)**

Created java properties filename as “application.properties”

**6. Testing**

Using Postman or any API testing tool to verify the functionality of the endpoints.

* **GET /todos**: Fetch all todos.
* **POST /todos**: Create a new todo.
* **GET /todos/{todoId}**: Fetch a todo by ID.
* **PUT /todos/{todoId}**: Update a todo by ID.
* **PATCH /todos/{todoId}**: Mark a todo as complete.
* **DELETE /todos/{todoId}**: Delete a todo by ID.
* **GET /todos/count**: Get the total count of todos.

**7. Documentation**

Swagger UI is set up correctly to document of API:

* Access Swagger UI at http://localhost:8080/swagger-ui.html to view and interact with the API documentation.

**8. Submission**

Package project as provided a link to Git repository containing the complete project.

By following these steps, I done a fully functional Todo application built using Spring Boot, with CRUD operations implemented and documented using Swagger UI.

creating RESTful web services with Spring Boot.