Create a Java web application that meets the following criteria:
- no presentation required
- no (bean) validation required
- compiles successfully
- supports at least 2 operations: adding and listing of items
- the operations are exposed either via an UI (server-side) or REST (JSON) or client-side UI + REST (JSON)
- items are stored in a database (one of: postgresql/mysql/mariadb/sqlite | mongodb | redis)
- uses one of the following technologies:
1. Spring MVC (with a mongodb or redis storage) – spring.io | start.spring.io
2. Vert.x – vertx.io | start.vertx.io
3. Spring WebFlux – spring.io | start.spring.io
4. Spark-Java – sparkjava.com
5. Micronaut – micronaut.io | micronaut.io/launch
6. Quarkus – quarkus.io
7. Helidon – helidon.io

Server-side UI templating:
- Thymeleaf
- FreeMarker
- Jade
- anything else except JSP

Some useful links:
- MongoDB
- Redis
- SQLite JDBC
- MongoDB mini-guide
- Redis Client

Guidance:
This is an exploratory exercise to check out various Java web technologies. A good place to start is by checking out the “quickstart” page. After going through the “get started”/”quickstart” page, you should end up with a basic web application (http server). At this point you can start looking into integrating with a database. You should also checkout the “docs” page for recommended clients/guides.

Warning: The listed Java technologies can have wildly different approaches on how things are done but by following the “quickstart” page you should end up with working examples:
- for Vertx, check out the “docs” page, they provide JDBC clients, Redis + Mongo Clients and also have guides for server-side templating. Beware though, Vert.x implies asynchronous code.
- Spring Webflux is more of a challenge because it implies reactive programming.
- Spark-Java – lambda focused
- Micronaut, Quarkus and Helidon provide both imperative/synchronous as well as async/reactive approaches. The first approach is closer to Spring MVC (easier because of familiarity)