The application was designed with Spring Boot. It follows the **tier architecture**.

All Extensions have been covered.

Apache Kafka was used as a distributed messaging means for “Extension 2 and 3”.

Apache Kafka was configured to allow creation of topics automatically.

The storage of the ID is a HashSet. Every duplicate addition makes no change to the state of the set. This is useful when multiple instances of the Spring Boot application are running.

At the beginning of each minute (using cron) the HashSet is cleared and the logs are sent to a separate topic (“countLog”) in Kafka.

The new IDs are broadcasted to all other listening instances (topic “distributedCounter”). This helps in synchronization among instances. However, the instances are not synchronized between them. This means that one instance could erase its HashSet before another, while both receiving a new id, resulting in a state where one instance has the counter larger than the other instance. This can be fixed with timestamping.

The endpoint always returns “ok”, and in case of failure a ControllerAdvice class handles the error and returns the “failed” message.

Tests have been written to assess basic request functionality.

Comments were added to explain certain methods and what they do.

Improvements in terms of configuration can be done. More application.properties entries could be useful and improve the overall design of the application.