UNIDAD TEMÁTICA 4 – PATRONES DE ARQUITECTURA – TRABAJO FINAL DE UNIDAD

Para este trabajo final de unidad deberán aplicar algunos de los patrones que hemos visto a la aplicación que han desarrollado en el trabajo final de la unidad anterior.

Además deberán incorporar la retroalimentación que hayan recibido de los profesores o de sus compañeros.

Elijan al menos dos patrones de cada uno de los grupos de patrones de disponibilidad, rendimiento y seguridad, y uno del grupo de patrones de facilidad de modificación y de despliegue; en total siete patrones en total a implementar. Si la aplicación está implementada con microservicios[1], pueden omitir patrones de facilidad de modificación y de despliegue; en ese caso, seis patrones en total a implementar.

1. Grupo de patrones de disponibilidad:
    - Bulkhead
    - Circuit Breaker
    - Competing Consumers
    - Event Sourcing
    - Health Endpoint Monitoring
    - Rate Limiting
    - Retry
    - Throttling
2. Grupo de patrones de rendimiento:
    - Cache-Aside
    - Queue-Based Load Leveling
    - Competing Consumers
    - Publisher/Subscriber
    - Asynchronous Request-Reply
    - CQRS
    - Materialized View
3. Grupo de patrones de seguridad:
    - Gatekeeper

[1] En el contexto de este trabajo cada contenedor deberá implementar un servicio resultante de una partición de dominio de primer nivel; es decir, no se acepta como microservicio una partición técnica implementada en contenedores.

- Federated Identity
- Gateway Offloading
- Sidecar
- Valet Key

4. Grupo de patrones de facilidad de modificación y de despliegue:
   - Deployment Stamps
   - External Configuration Store
   - Messaging Bridge
   - Strangler Fig
   - Blue/green deployment.
   - Rolling upgrade.
   - Canary testing.
   - A/B testing.

Parte 1

El entregable asociado a esta parte de la TFU, por cada grupo será un documento donde se incluya la siguiente información:

- Modelos en notación UML en el que pueda verse la aplicación del patrón. Podrán usar diagramas de clases, de secuencia, de componentes o de despliegue.
- Una explicación de cómo el patrón logra el atributo de calidad correspondiente y de cómo pueden comprobarlo mediante pruebas empíricas.

Parte 2

La aplicación debe ser una API REST que se pueda probar usando curl o Postman.

La aplicación debe ser desplegada y ejecutada en Docker o en Azure Virtual Machines para que pueda funcionar independientemente del ambiente en la que la probaremos los profesores.

Los entregables asociados a esta parte de la TFU de cada grupo serán:

- El **código de la aplicación** que implemente los conceptos mencionados. Pueden usar cualquiera de los lenguajes y frameworks que han visto hasta ahora en la carrera.
- El archivo **docker-compose.yaml** para desplegar y ejecutar la aplicación si usan Docker o un **archivo de secuencia de comandos .azcli** para desplegar y ejecutar la aplicación si usan Azure Virtual Machines.

- Los **scripts** para probar cómo los patrones implementados logran que la aplicación tenga los atributos de calidad correspondientes.