SEGUNDO PARCIAL SOFTWARE II

Presentado por:
Jesus David Cardenas Sandoval
Jesus Gabriel Parra Dugarte
Elkin Ariel Morillo Quenguan

Presentado a:
Julio Ariel Hurtado Alegria
Ricardo Zambrano

Ingeniería de Software II

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Programa de Ingeniería de sistemas

Popayán, Cauca

10/Noviembre/2023

Historias Épicas Mercado de valores

- **HE_001 Gestión de acciones**
  - HU001- Creación de acciones
  - HU002- Edición de acciones
  - HU003- Eliminación de acciones
  - HU004- Edición del precio una acción
  - HU005- Consultar todas las acciones
  - HU006- Filtro de acciones por id

Historias de usuario y criterios de aceptación

Mercado Valores HU/CA

Atributos de calidad

Atributos de calidad - Bolsa de valores

Backlog

Esquema del Backlog

Repositorio

Link GitHub

Tácticas de Escalabilidad

Retención del desempeño: Para lograr la táctica de retención del desempeño, tenemos la arquitectura de cliente-servidor y publicador-suscriptor, pues cada uno ayuda a mantener un alto nivel de rendimiento, mejorar la capacidad de del sistema y el manejo de la concurrencia, además en cliente-servidor el trabajo puede ser distribuido y en publicador-suscriptor nos ofrece mantener un rendimiento óptimo en el sistema a largo plazo.

Rentabilidad: En este caso tenemos la arquitectura orientada a servicios que permite desplegar servicios de forma independiente, ofreciendo como resultado mejorar la eficiencia en el sistema , de la misma manera tenemos la arquitectura de microservicios, cada servicio puede ser desplegado en su servidor virtual, siendo ideal para virtualizar los recursos del servidor.

Tácticas de Modificabilidad

Reducir acoplamiento: Las arquitecturas que mejor se adaptan, es la de capas y la de microkernel, cada capa es independiente y la comunicación entre las capas es a través de interfaces, dando como resultado lo que buscamos, minimizar las dependencias entre los módulos y el microkernel, cada plugin es independiente entre sí, cada uno solo se conecta al núcleo del sistema, ofreciendo un acoplamiento mínimo y reduciéndolo.

Reducir el tamaño de un módulo: Las arquitecturas que se pueden considerar son la de modelo-vista-controlador, siendo ideal para separar un módulo en varios módulos más pequeños y permitiendo una modificación de manera independiente, y la de microservicios, cada microservicio es pequeño y se encarga de una funcionalidad, de tal manera que se puede reducir el modulo en varios microservicios con una única responsabilidad.

Atributo a descomponer : Escalabilidad

Perspectivas

Perspectiva de componentes
La perspectiva de componentes se enfoca en descomponer el sistema en módulos o componentes y la interacción entre ellos, de tal manera que que se puedan modificar sin afectar a los demás y al estar distribuidos mejoren la capacidad de respuesta del sistema, un patrón arquitectónico que facilita esta perspectiva, es el de microservicios , ya que permite que partes específicas del sistema se desarrollen e implementen de manera independiente o individual facilitando la escalabilidad en que cada componente.

Perspectiva de módulos
La perspectiva de módulos, tiene un enfoque sobre la organización modular y evaluar cómo se comunican los módulos, la arquitectura publicador-suscriptor nos ofrece una comunicación asíncrona, además una gran escalabilidad en cada módulo y desacoplamiento, cada módulo se debe interesar a eventos específicos de tal manera que tendrá solo una responsabilidad y se encargará de ella, ofreciendo una gran escalabilidad en el sistema y una independencia entre módulos para permitir el desacoplamiento entre ellos.

Perspectiva de localización
La perspectiva de localización se centra en la organización espacial y disposición física de los elementos, por lo tanto , el patrón arquitectónico a utilizar será n-tier, dicho patrón nos permite desplegar las capas en ubicaciones geográficas específicas,y nos brinda utilidad para optimizar el rendimiento y latencia ubicando dichas capas donde se necesite un procesamiento más intensivo.

MODELO C4

Contexto

En el primer nivel tenemos el contexto, en el cual identificamos nuestro sistema principal este sistema será una aplicación de seguimiento del mercado de valores , centrándonos en el backend de la aplicación.

DiagramaC4-Diagrama de Contexto.drawio.png

Contenedor

En el segundo nivel encontramos dos contenedores: la aplicación de gestión de acciones, que se realizará en Java Spring Boot y una base de datos relacional que inicialmente será MYSQL.

DiagramaC4-Diagrama de Contenedor.png

Componente

La arquitectura del sistema estará basada en capas, por lo tanto en este tercer nivel encontramos la gestión de acciones dividida en varias capas: modelo, controlador, excepciones, repositorio y servicio. Cada capa actúa como un componente en nuestro sistema y tienen una funcionalidad específica.

DiagramaC4-Diagrama de Componentes.drawio.png

Clases

En este último nivel tenemos un detalle sobre todas las clases que tiene el sistema, muestra cada clase , sus atributos y métodos y las relaciones entre ellas.

DiagramaDeClasesUML.png

Diagrama de secuencia

Diagrama de secuencia Registrar Accion.png

Diagrama de Módulos

Diagrama_Modulos.drawio.png

Diagrama de componentes

Diagrama_Componentes.drawio.png