Arquitectura de Software

Laboratorio 2

Nicolas Daniel Vargas Ortiz

Juan Camilo Merchán Loza

Pontificia Universidad Javeriana

Ingeniería de Sistemas

Octubre 31 de 2022

Marco Conceptual

Para el desarrollo de este laboratorio se tuvo que investigar sobre los siguientes conceptos:

- Jakarta: Jakarta EE es un conjunto de componentes de software, API, para desarrollar aplicaciones Java específicamente empresariales. Estos componentes a menudo se denominan especificaciones.
- Swagger: es un conjunto de herramientas de código abierto creadas en torno a la especificación OpenAPI que puede ayudarlo a diseñar, crear, documentar y consumir API REST.
- Payara: Payara Server es un servidor de aplicaciones de código abierto derivado de GlassFish. Fue creado en 2014 por C2B2 Consulting como reemplazo directo de GlassFish después de que Oracle anunciara que suspendería el soporte comercial para este.
- MariaDB: es uno de los servidores de bases de datos más populares del mundo que convierte los datos en información estructurada. MariaDB se usa debido a que es rápido, escalable y robusto, con un rico ecosistema de motores de almacenamiento, complementos y muchas otras herramientas que lo hacen muy versátil para una amplia variedad de casos de uso.
- REST: Una API REST (también conocida como API RESTful) es una interfaz de programación de aplicaciones (API o API web) que se ajusta a las restricciones del estilo arquitectónico REST y permite la interacción con los servicios web RESTful. REST significa transferencia de estado representacional y fue creado por el científico informático Roy Fielding.
- Java Enterprise Edition: es el estándar en software empresarial impulsado por la comunidad. Java EE se desarrolla utilizando el proceso de la comunidad de Java, con contribuciones de expertos de la industria, organizaciones comerciales y de código abierto, grupos de usuarios de Java e innumerables personas. Cada versión íntegra nuevas características que se alinean con las necesidades de la industria, mejora la portabilidad de las aplicaciones y aumenta la productividad de los desarrolladores.
- CRUD: CRUD es el acrónimo de CREAR, LEER, ACTUALIZAR y ELIMINAR. Estos términos describen las cuatro operaciones esenciales para crear y administrar elementos de datos persistentes, principalmente en bases de datos relacionales y NoSQL.
- Arquitectura de capas: La arquitectura en capas consta en dividir la aplicación en capas, con la intención de que cada capa tenga un rol muy definido, este estilo arquitectónico no define cuantas capas debe de tener la aplicación, sino más bien, se centra en la separación de la aplicación en capas.

Procedimiento

Para el desarrollo del laboratorio, lo primero que hicimos fue configurar los componentes de manejo de bases de datos, es decir, el MariaDB y el Payara, en nuestro caso dockerisamos estas dos herramientas para evitar instalar en los computadores de la universidad y tener acceso a ellos desde cualquier computador donde trabajáramos. Luego de la instalación y configuración, se cargo el archivo SQL correspondiente de la base de datos persona_db usando HeidiSQL y se crea un usuario nuevo con las credenciales de “personapp” y con contraseña persona123 y despues procedemos a otorgarle permisos de gestión sobre la base de datos.

Despues de esto en Visual Studio Code se instalaron las extensiones que nos permitieran hacer uso de las herramientas ya instaladas, despues de eso se inicia el servidor de payara, se abre la terminal de administración y se crea un pool usando los datos con el usuario creado previamente. Posterior a esto, se crea el proyecto de jakarta y se configura con los valores de la base de datos y el puerto en donde se montó.

Una vez habiendo terminado de crear el proyecto en ´personapp web en paquete de servicios se crean dos nuevas clases, personasResources y profecionesResource en donde cada una se definieron los endpoints de cada una para su correspondiente CRUD. Debido a las relaciones bidireccionales de algunas entidades se tuvo que crear DTOs para cada una para su correcta conversión a formato JSON y en cada “resource” se implementaron las respectivas funciones para volver una entidad un DTO y viceversa. Se le hizo “install” al proyecto con Maven y el resultante archivo EAR se desplego en una instancia dockerizada de payara server. Se le realizaron pruebas a todos los endpoints usando la herramienta de postman.

Para la implementación de swagger al proyecto se incluyeron dos librerías en el archivo Pom de personapp-ear, estas dependencias son las de microprofile que se encarga de crear un documento con todos los endpoints del proyecto y openapi-ui que se encarga de tomar esta información y mostrarla en una interfaz swagger

Conclusiones y Lecciones Aprendidas

Para terminar, podemos concluir que para el correcto desarrollo de este laboratorio es de suma importancia la correcta implementación y manejo de todas las herramientas sobre todo de las que interviene en el manejo de los datos, cosas como MariaDB y Payara son muy útiles sin embargo su configuración puede ser algo tediosa y complicada pues el tener que descargar cosas que requieren por separado hace que sea confuso y se puedan generar errores solo por guardar cosas en carpetas distintas a donde estos programas lo esperan, cosa que creemos ya debería tener en el momento de la

descarga, sin embargo, estas aplicaciones facilitan bastante agilizan bastante el manejo y gestión de la información.

Ciertamente en el ámbito del manejo y creación de bases de datos jakarta y mariaDB brindan muchas herramientas de gestión y control que facilitan las tareas sobre la información y aprender a manejar y configurar estas dos es de gran importancia ya que al ser un estándar al trabajar con javaEE es muy utilizado en la industria y de seguro nos servirá para abrirnos paso en el campo laboral.

Referencias

- https://mariadb.com/
- https://www.payara.fish/
- https://netbeans.apache.org/download/index.html
- https://www.arquitecturajava.com/jakarta-ee/
- https://swagger.io/
- https://blog.payara.fish/jakarta-ee-java-ee-guide
- https://www.oracle.com/java/technologies/java-ee-glance.html