ARQUITECTURAS WEB 2021

Trabajo Práctico 5: “Documentación y Testing”

“alumnos”: { [Bellido Sebastián, Cepeda Tomás, Nuñez David] }

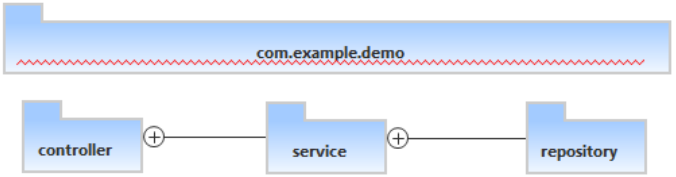
UNICEN / FCE / TUDAI

INTRODUCCIÓN

En el presente informe se tiene como objetivo describir el diseño de la aplicación. Cabe aclarar que solo se hará mención de las clases y paquetes relacionados con la arquitectura de la aplicación.

ARQUITECTURA GENERAL

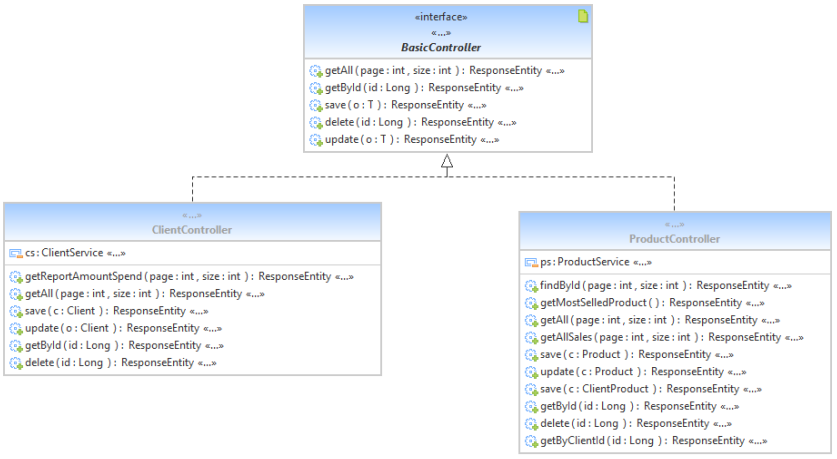
Con base en el framework de Spring y en la problemática plantea en el TP4 se diseñó una aplicación web, con una arquitectura de 3 capas: capa de controladores, capa de servicios y capa de repositorios:



Capa de Controlador:

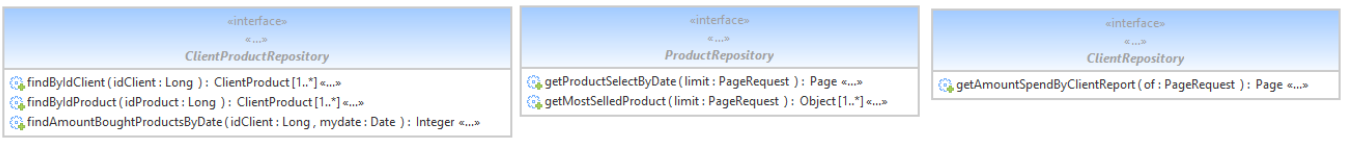
La interfaz *BasicController* define los métodos comunes para todos los controladores.

Esta capa es la encargada de manejar las peticiones HTTP con los clientes. Se encuentra en el paquete de "com.example.demo.controller" e intercambia peticiones HTTP con los distintos clientes. Mientras que interactúa con la capa de servicios, invoca los métodos de los mismos para resolver las peticiones de los clientes.



Capa de Persistencia:

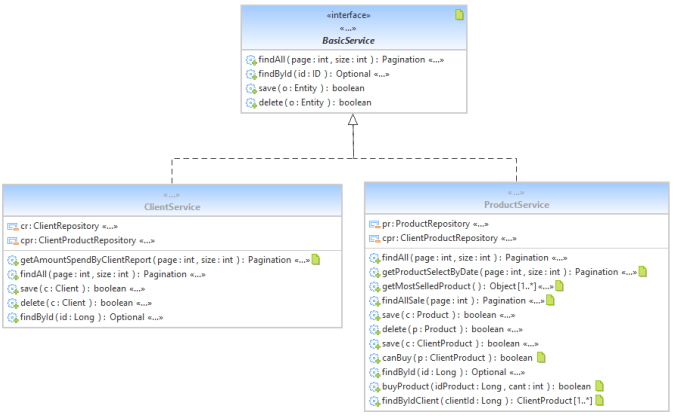
En esta capa se implementan los métodos CRUD (Create, Read, Update, Delete)sobre las entidades en la DDBB. Se encuentra en el paquete de "*com.example.demo.repository*". Se decidió utilizar el Patrón Repositorio para poder guardar los datos en DDBB. Además, contiene las consultas que serán utilizadas por las capas superiores para proveer los distintos servicios que ofrece la aplicación.



Capa de Servicios:

Esta capa es la encargada de manejar la lógica del negocio. Se encuentra en el paquete de “com.example.demo.services” y utiliza los métodos de los repositorios para poder realizar todas las operaciones relacionadas a esta lógica.

La interfaz *BasicService* define los métodos comunes para todos los servicios.



POJO’s o Modelos:

Del análisis de la problemática se identificaron 3 entidades distintas: *Cliente*, *Producto*, y *ClienteProducto*, esta última representa las compras realizadas por los clientes. Después del análisis, se realizó el Diagrama de Entidad de Relación y el correspondiente Diagrama de Clases representando las distintas entidades. Adicionalmente, se implementó el Patrón Data Transfer Object, para que los repositorios puedan utilizar este patrón (DTO) en las respuestas de sus consultas.

La interfaz *EntityPojo* define un método común para todas las entidades.

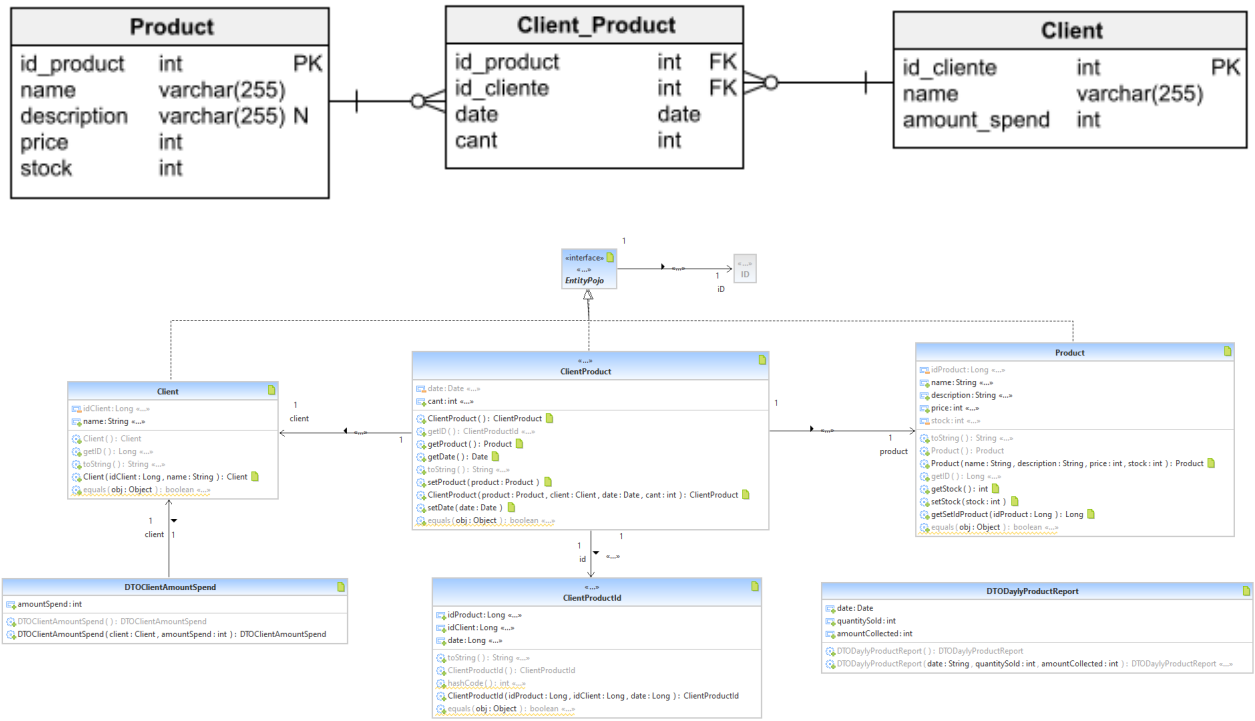
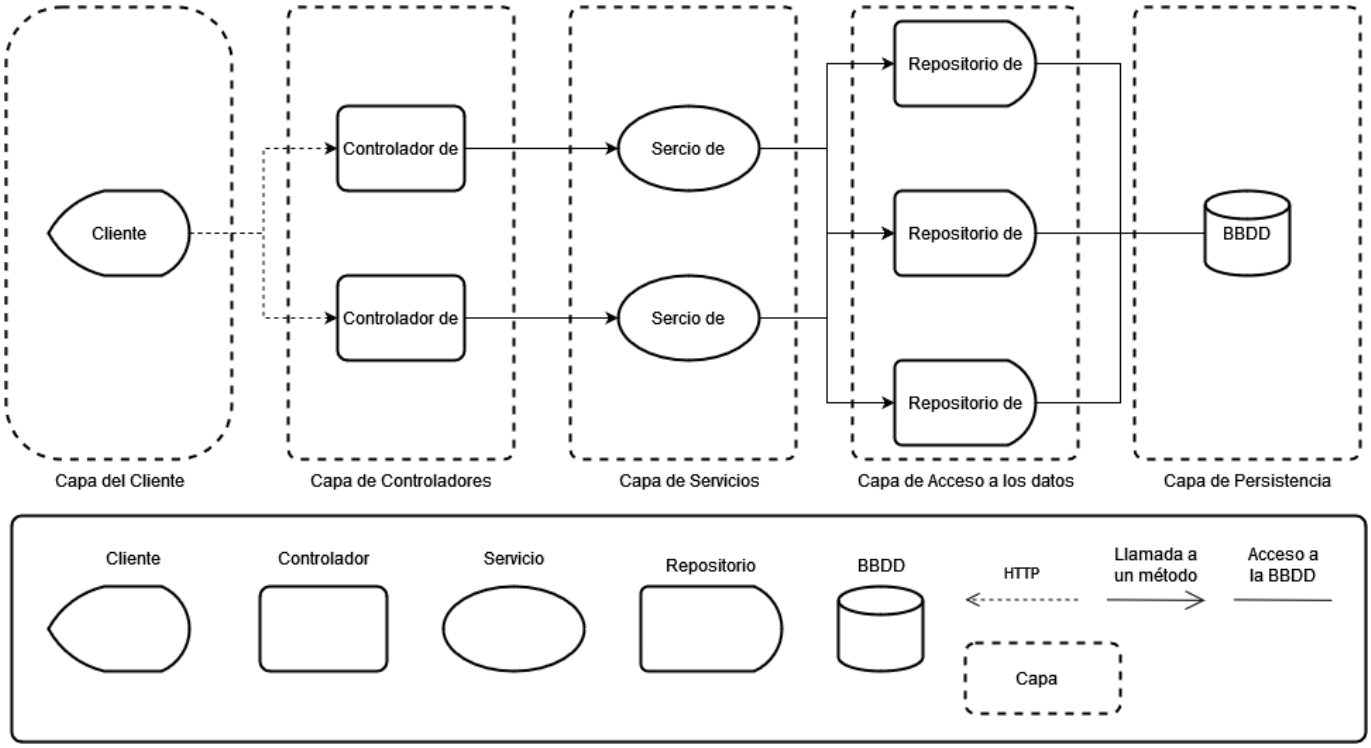


Diagrama de Componentes y Conectores

En este diagrama se puede ver cómo se estructura en términos de un conjunto de elementos que poseen comportamiento en tiempo de ejecución e interacciones con otros elementos. Se omitió el diagrama de Vista de Asignación, dado que todos los componentes corren en el mismo nodo.



Despliegue de aplicación en PAAS

Por último, se realizó un despliegue en la plataforma de Heroku, el cual se puede ver en el siguiente link: https://aweb-tp5.herokuapp.com/