Mario Fernández Suárez

I.E.S. Castelar

OrganList

Proyecto final de D.A.M.

Contenido

[Descripción 1](#_Toc86166999)

[Herramientas 1](#_Toc86167000)

[Diagrama E-R 2](#_Toc86167001)

[Esquema BBDD 3](#_Toc86167002)

[Repositorios 3](#_Toc86167003)

[Objetivos 4](#_Toc86167004)

[API (Application Programming Interfaces) 4](#_Toc86167005)

[OpenJDK 4](#_Toc86167006)

[Maven 5](#_Toc86167007)

[Definición y creación del arquetipo 6](#_Toc86167008)

[Bibliografía 7](#_Toc86167009)

# Descripción

Este proyecto, en general, trata sobre una aplicación web reactiva, crearemos una API, documentada con swagger-ui (OpenAPI 3), que gestiona la lista de la compra, pudiendo esta contener productos de distinta índole y dando la posibilidad de convertirla en una lista de cosas que hacer, que se podrán ir tachando.

En dicha lista se dará la opción de realizar la compra ayudándonos de la API de un supermercado, pudiendo guardar las diferentes listas, asociarles un nombre, descripción o pudiendo añadirlas a una lista de favoritos y dando la opción de compartirlas por Whatsapp.

El ejemplo más es el de un supermercado y sus productos, pero los productos tendrán una propiedad que indique de la categoría a la que pertenecen, pudiéndose usar en varios ámbitos.

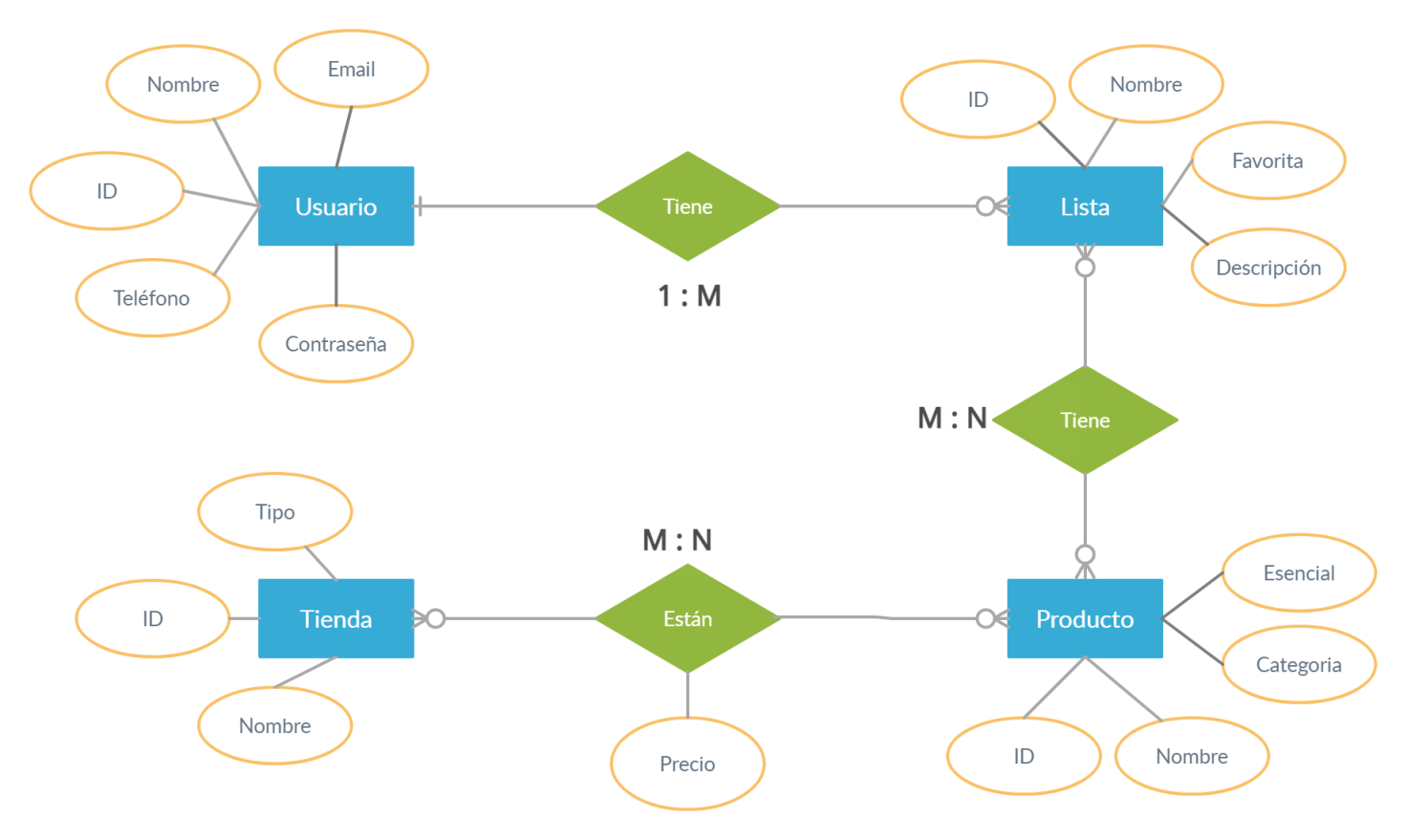
Los productos también tendrán una propiedad que indique sin son productos esenciales en dicha lista, es decir, productos que no puedan faltar.

# Herramientas

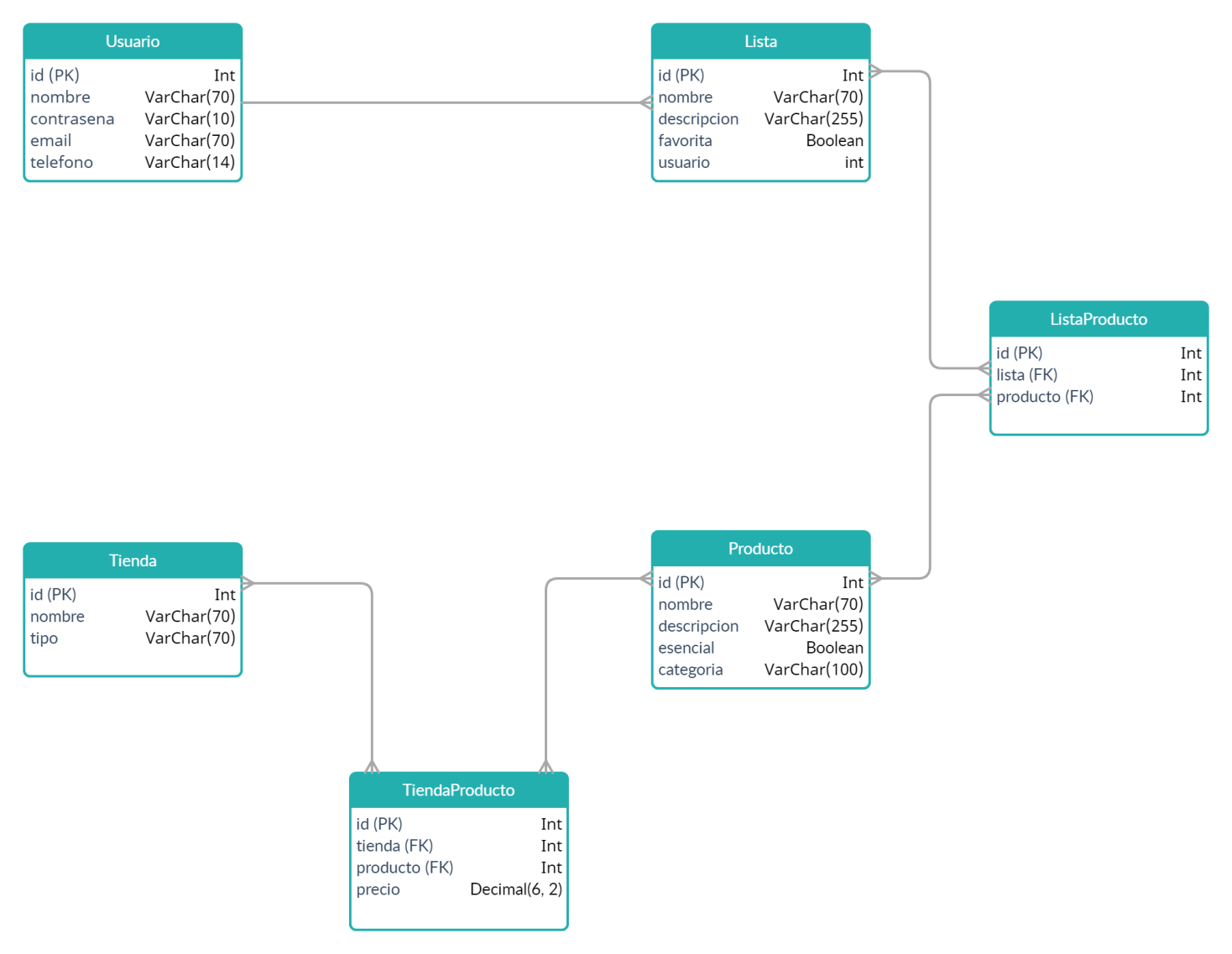
* **Visual Studio Code**: es un editor de código fuente, compatible con varios lenguajes de programación, con características y extensiones disponibles que facilitan el desarrollo web.
* **IntelliJ IDEA Community Edition**: es un entorno de desarrollo integrado (IDE), desarrollado por JetBrains, cuya versión gratuita está orientada para desarrollos java, permitiendo desarrollos en kotlin y Android y cuya versión de pago permite el desarrollo con tecnologías de la parte front.
* **Spring Boot**: framework java orientado al desarrollo web, en el cual utilizaremos una programación reactiva mediante la librería WebFlux. Este IDE contiene un servidor Tomcat embebido que nos permitirá arrancar la aplicación. Utilizando la última versión de java libre (OpenJDK 17).
* **MariaDB (MySQL):** es un sistema de base de datos relacional de código abierto .
* **DBeaver:** es un gestor de BBDD multiplataforma.
* **OpenApi 3**:  es una especificación que define un formato de descripción abierto e independiente de los fabricantes para los servicios de API sirve para describir, desarrollar, probar y documentar las APIs de los servicios web RESTful.
* **Angular**: plataforma de desarrollo construida en TypeScript, basado en componentes para crear aplicaciones web escalables. Lo utilizaremos en su última versión hasta el momento, la v12.
* **Postman**: software para crear peticiones HTTP y poder probar las APIs, pudiendo definir colecciones de las mismas y diferentes entornos con la definición de sus propias variables, entre otras cosas.
* **Maven:** es una herramienta de construcción y organización de proyectos.Lo utilizaremos en su última versión hasta el momento, la v3.8.
* **Git/GitHub**: GitHub es una web para alojar proyectos utilizando el sistema de control de versiones Git.

# Diagrama E-R

Este diagrama lo he digitalizado con ayuda de la aplicación web de Creately.



# Esquema BBDD



# Repositorios

Las versiones/desarrollos de esta aplicación se subirán a los siguientes repositorios públicos de [GitHub](https://github.com/MarioFS94):

* [Parte BackEnd](https://github.com/MarioFS94/OrganList-Back)
* [Parte FrontEnd](https://github.com/MarioFS94/OrganList-Front)

# Objetivos

Los objetivos que tengo con este proyecto es hacer una aplicación incluyendo:

* La generación del proyecto a partir de un arquetipo Maven.
* Implementar todo el flujo de desarrollo, con librerías reactivas y siguiendo el patrón redux en la parte front con Akita en Angular.

# API

Una API (Application Programming Interfaces) es un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones. API significa interfaz de programación de aplicaciones.

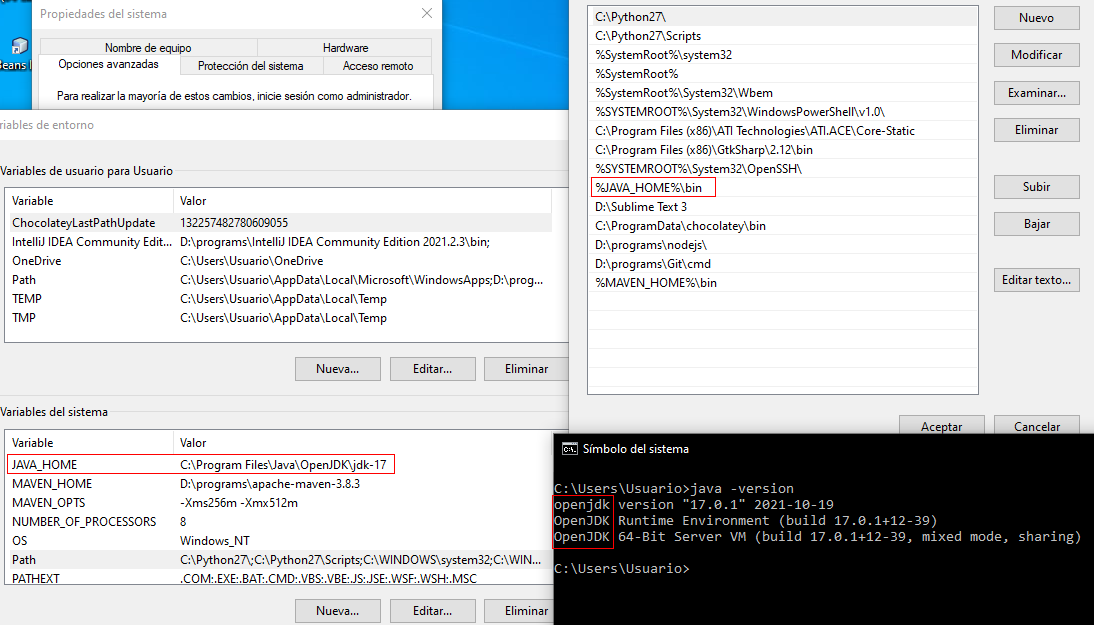


Las API permiten que sus productos y servicios se comuniquen con otros, sin necesidad de saber cómo ni en que lenguaje están implementados. Esto simplifica el desarrollo de las aplicaciones y permite ahorrar tiempo y dinero. Las API le otorgan flexibilidad; simplifican el diseño, la administración y el uso de las aplicaciones, y proporcionan oportunidades de innovación, lo cual es ideal al momento de diseñar herramientas y productos nuevos (o de gestionar los actuales).

# OpenJDK

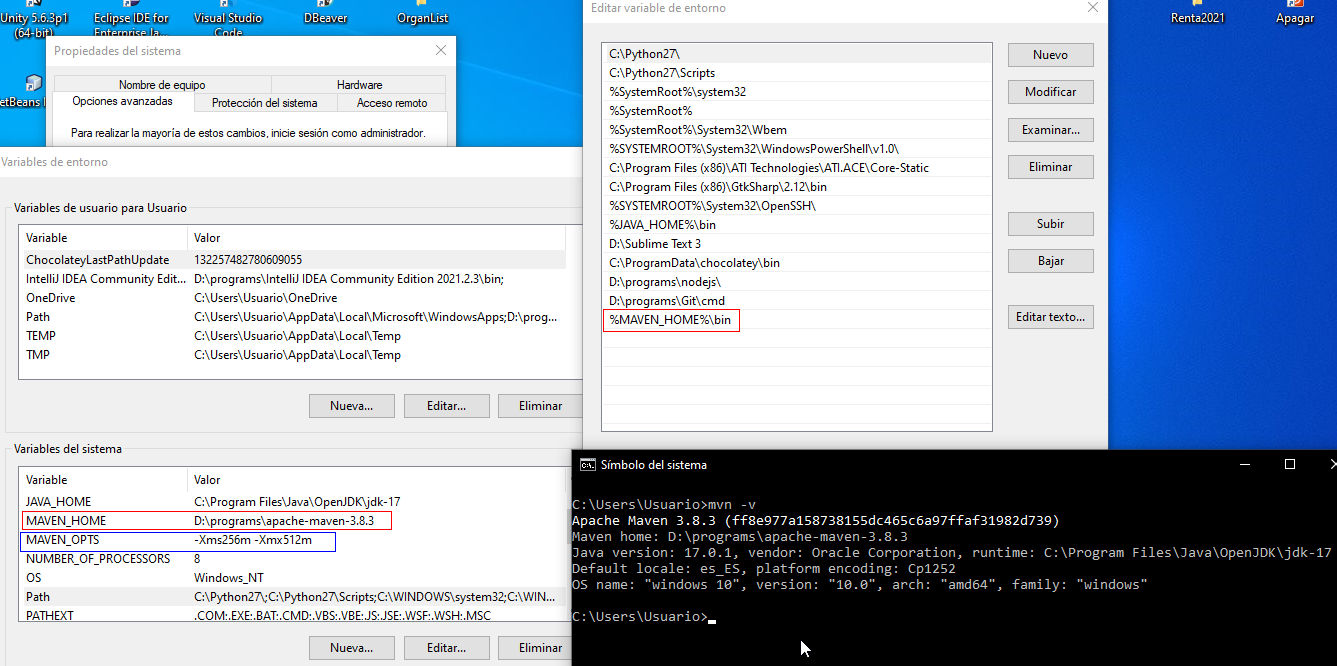
Java a partir de su versión 11, se ha hecho de pago, para aquellas aplicaciones que se quieran subir a producción con dicho jdk. Sin embargo, Oracle también ha sacado una versión open source, OpenJDK, para poder usar las versiones superiores necesidad de pagar.

Para su configuración he modificado y borrado variables del sistema, quedando así:



# Maven

Maven es una herramienta de construcción y organización de proyectos, que tiene el objetivo de simplificar los procesos de build (compilar y generar ejecutables a partir del código fuente).



La variable de MAVEN\_OPTS no es necesaria configurarla, indica que capacidad tendrá en la memoria disponible para proyectos java.

# Definición y creación del arquetipo

En esencia, un arquetipo de Maven es un conjunto de herramientas para crear plantillas para proyectos. Un arquetipo se define como una estructura o modelo original a partir del cual se hacen todas las demás cosas del mismo tipo. Los arquetipos ayudarán a los autores a crear plantillas de proyectos Maven para los usuarios, proporcionando a los usuarios los medios para generar versiones parametrizadas de esas plantillas de proyectos.

El uso de arquetipos proporciona a los desarrolladores una manera rápida y consistente, con las mejores prácticas, para crear un proyecto con unas determinadas clases y organización de paquetería, a partir de una serie de ficheros predeterminados.

También permite que el mecanismo de arquetipos sea aditivo, y con ello nos referimos a permitir que partes de un proyecto sean guardadas para un determinado arquetipo, de modo que piezas, componentes o aspectos de un proyecto puedan ser añadidos a proyectos ya existentes.

Si ejecutamos el comando:

**mvn archetype:generate**

Con este comando descargaremos todos los arquetipos configurados en el repositorio, para descargar uno en concreto ejecutaremos:

**mvn archetype:generate -Dfilter=[archetype name]**

Los arquetipos predefinidos por Maven son los siguientes:

|  |  |
| --- | --- |
| **Archetype ArtifactIds** | **Description** |
| maven-archetype-archetype | An archetype to generate a sample archetype project. |
| maven-archetype-j2ee-simple | An archetype to generate a simplifed sample J2EE application. |
| maven-archetype-mojo | An archetype to generate a sample a sample Maven plugin. |
| maven-archetype-plugin | An archetype to generate a sample Maven plugin. |
| maven-archetype-plugin-site | An archetype to generate a sample Maven plugin site. |
| maven-archetype-portlet | An archetype to generate a sample JSR-268 Portlet. |
| maven-archetype-quickstart | An archetype to generate a sample Maven project. |
| maven-archetype-simple | An archetype to generate a simple Maven project. |
| maven-archetype-site | An archetype to generate a sample Maven site which demonstrates some of the supported document types like APT, XDoc, and FML and demonstrates how to i18n your site. |
| maven-archetype-site-simple | An archetype to generate a sample Maven site. |
| maven-archetype-webapp | An archetype to generate a sample Maven Webapp project. |

En la ejecución nos hará varias preguntas, con sus respectivas sugerencias, para poder identificar nuestro proyecto y rellenar nuestro pom.xml. Nos preguntará por el **groupId**, que es el grupo mediante el cual identificaremos nuestro proyecto, se suele definir como la estructura de un dominio o si ya existe nuestra página, con el nombre de nuestro dominio invertido, por ejemplo:

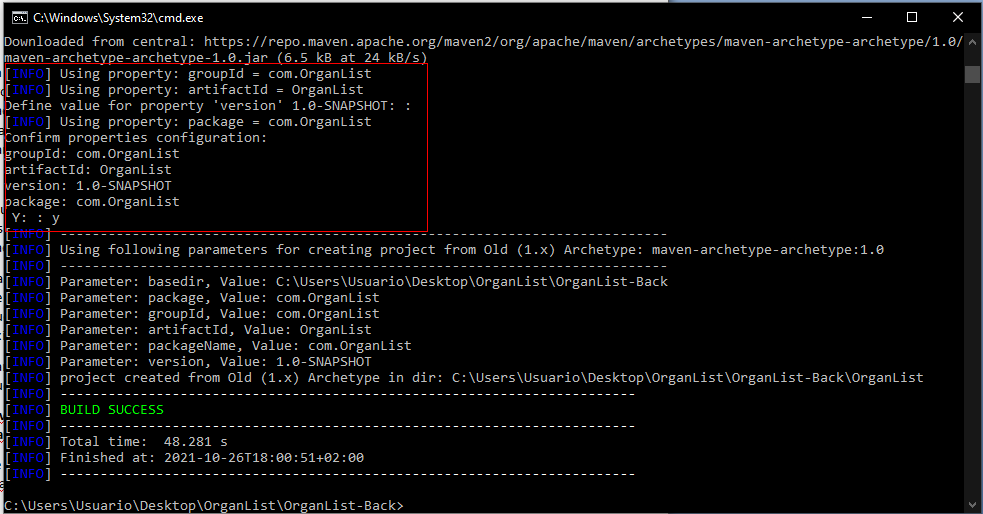
Imaginemos nuestro dominio, organlist.com -> com.organlist, sería nuestro groupId.

También nos preguntará por el **artifactId**, que sería el nombre de nuestro proyecto.

Para crear nuestro arquetipo, nos situaremos en la carpeta donde queramos que este alojado nuestro proyecto y ejecutaremos el siguiente comando en la consola:

**mvn archetype:generate -DgroupId=com.organList -DartifactId=organlist -DarchetypeArtifactId=maven-archetype-archetype**

Se descargarán las dependencias contenidas en nuestro arquetipo del repositorio central de maven[**https://repo.maven.apache.org/maven2/**](https://repo.maven.apache.org/maven2/)y hará ciertas preguntas, que podremos configurar para que se rellene el pom.xml, terminando la ejecución del comando así:



2964: remote -> tech.ibit:ibit-springboot-web-starter-archetype (ibit-springboot-web-starter-archetype)

# Bibliografía

* <https://app.creately.com/>
* <https://maven.apache.org/guides/introduction/introduction-to-archetypes.html>
* <https://maven.apache.org/guides/mini/guide-creating-archetypes.html>
* <https://www.adictosaltrabajo.com/2008/06/09/creararquetiposmaven/>
* <https://enmilocalfunciona.io/por-que-utilizar-arquetipos-maven/>
* <https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/que-es-openapi/>