Actividad SpringBoot

Requisitos

Documento			
Proyecto	Actividad SpringBoot	Código proyecto	No aplica
Título	Requisitos	Versión	1.0
Autor			
Aprobado por			
Fecha aprobación		Fecha publicación	No aplica
Clasificación			

Historial de cambios					
Versión	Fecha	Motivo	Puntos		
			Modificados	Añadidos	Eliminados
1.0	08/06/2022	Creación del documento			
1.0	08/06/2022	Definición del alcance			
1.0	13/06/2022	Revisión y aprobación			

Índice

1. Objetivo

El objetivo del documento es la presentación del guion de una práctica con la intención de identificar los conocimientos sobre el desarrollo de back-end Java y Springboot del técnico que la desarrolle.

2. Alcance

El objetivo es exponer un guion para que el técnico pueda desarrollar una pequeña aplicación cuya funcionalidad será la gestión de una página de vacantes de empleo.

3. Glosario de términos y definiciones

Termino	Definición
API	Una API REST define un conjunto de funciones que los desarrolladores pueden realizar solicitudes y recibir respuestas a través del protocolo HTTP, como GET y POST.
Endpoint	Es una URL que reciben o retornan información de un API.
Backend Rest	Es una pieza de desarrollo software que se encarga de la lógica de negocio, seguridad y acceso a datos (como partes más importantes). Expone un interfaz a modo de API Rest para que una aplicación Web muestre la información a los usuarios e interactúen de forma visual.

4. Documentos relacionados

Código	Documento
https://app.swaggerhub.com/apis/NobelMedia/NobelMasterData/2.1#/default	API público de los premios Nobel

5. Requisitos técnicos

El Backend Rest se deberá construir con una versión de Spring Boot 2.6.x o 2.7.0, y una versión 11 de Java (recomendable open jdk-11.0.2).

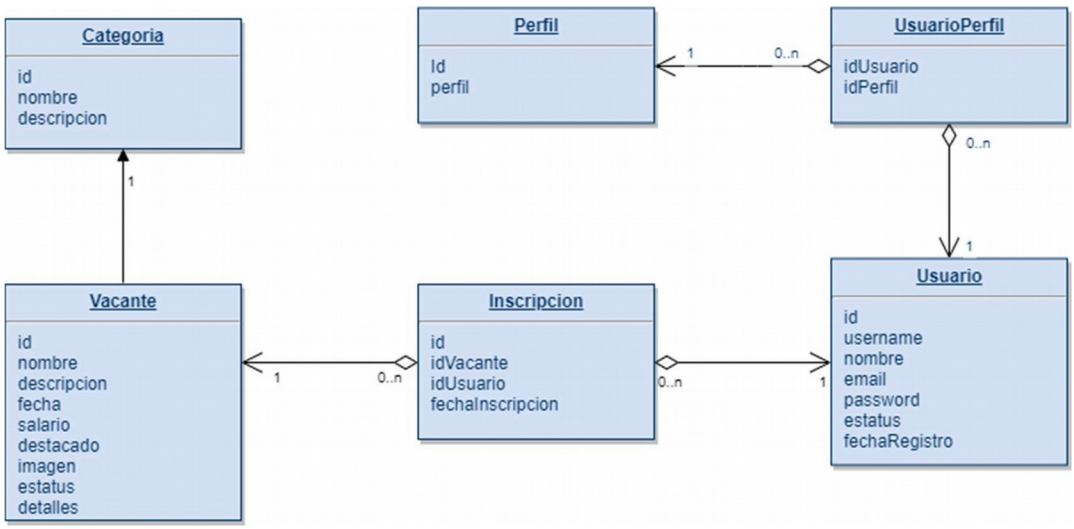
No existe ningún otro requisito en cuanto las tecnologías y/o frameworks a utilizar para l construcción del Backend.

Se recomienda el uso de un BBDD embebida (por ejemplo, H2) para facilitar el desarrollo posteriores pruebas, evitando la instalación de motores de BBDD externos.

6. Requisitos funcionales

La aplicación para la que se debe construir el Backend Rest es una página Web que ofre vacantes de empleo.

El diagrama de BBDD UML que debe seguir es el siguiente:



La aplicación deberá de cumplir, al menos, la siguiente funcionalidad:

6.1 Entidad Usuarios

- Se deben implementar los endpoint básicos que permitan la gestión de la entidad.
  - Obtener todos los usuarios
  - Obtener un usuario por su identificador
  - Crear un nuevo usuario
  - Modificar un usuario existente
  - Borrar un usuario existente

- Búsqueda de usuarios por varios parámetros.

- En el acceso a datos para la gestión de esta entidad se pide la creación de un repositorio JPA que haga uso de los siguiente:
  - Métodos por defecto del interfaz heredado. Métodos de consulta simples y métodos de consulta por palabras clave.
  - Query methods
  - Uso de ExampleMatcher y Matcher en el controlador para la búsqueda de usuarios.

6.2 Entidad Perfil

- Se deben implementar los endpoint básicos que permitan la gestión de la entidad.
  - Obtener todos los perfiles
  - Obtener un perfil por su identificador
  - Crear nuevos perfiles
  - Modificar un perfil existente
  - Borrar un perfil existente
- Esta entidad está relacionada con la entidad Usuario en una relación N:N. Hacer uso de JoinTable para la creación de tabla intermedia que las relacione.

6.3 Entidad Categoría

- Se deben implementar los endpoint básicos que permitan la gestión de la entidad.
  - Obtener todas las categorías
  - Obtener una categoría por su identificador
  - Crear una nueva categoría
  - Modificar una categoría existente
  - Borrar una categoría existente

6.4 Entidad Vacante

- Se deben implementar los endpoint básicos que permitan la gestión de la entidad.
  - Obtener todas las vacantes
  - Obtener una vacante por su identificador
  - Crear una nueva categoría
  - Modificar una vacante existente
  - Borrar una vacante existente
  - Búsqueda de vacantes por varios parámetros.
- En el acceso a datos para la gestión de esta entidad se pide la creación de un repositorio JPA que haga uso de los siguiente:
  - Métodos por defecto del interfaz heredado. Métodos de consulta simples y métodos de consulta por palabras clave.
  - Query methods
  - Uso de ExampleMatcher y Matcher en el controlador para la búsqueda de vacantes.
- Esta entidad está relacionada con Categoría en una relación 1:1

6.5 Validaciones

Se deben realizar validaciones sobre los distintos campos de las entidades. Se deben incluir al menos validaciones de obligatoriedad, de tamaño máximo y de formatos.

Procesar los errores de validación para mostrarlos en el resultado de la llamada al endpoint.

6.6 Tratamiento de respuestas

Los endpoint implementados tienen que devolver como respuesta un objeto de tipo ResponseEntity para manipular la respuesta HTTP. Se pueden consultar los distintos códigos de respuesta aquí: https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

6.7 Añadir seguridad con spring-boot-security

Hacer uso de Spring-boot-security para securizar la aplicación.

- Gestionar los accesos a determinados endpoints con usuario autenticado. Dejar otros endpoint para el acceso público.
- Crear dos perfiles 'ADMIN' y 'USER' y controlar el acceso a los endpoints securizados en función del perfil.
- Crear controlador con endpoint para el login de un usuario. Añadir la lógica necesaria para crear un token jwt (Json Web Token) para el acceso de los usuarios.

6.8 Acceso a API externa

Se debe construir un cliente en nuestro Backend Rest que sea capaz de realizar llamadas a un Backend Rest externo. La API pública que podemos utilizar para realizar este punto es la siguiente:

- https://app.swaggerhub.com/apis/NobelMedia/NobelMasterData/2.1#/default

Concretamente el endpoint a utilizar es:

- http://api.nobelprize.org/2.1/nobelPrize/{category}/{year}

Se pide realizar un cliente que sea capaz de realizar llamadas correctas a este endpoint, recuperar los resultados y gestionar los errores.

Se permite, si se prefiere, el uso de otra API pública distinta, o la invocación a otros métodos distintos de esta misma API.

6.9 Integración REST API con documentación Swagger UI

Integrar en la aplicación Swagger UI para crear la documentación del servicio REST, y po realizar pruebas desde esta herramienta. (https://swagger.io/tools/swagger-ui/)

6.10 Pruebas unitarias con SpringBootTest

Realizar pruebas unitarias de algunos de los endpoints haciendo uso de SpringBootTest. Lo endpoint a probar quedan a decisión del candidato.

6.11 Creación de colección de pruebas con Postman

Hacer uso de la herramienta Postman para crear una batería de pruebas de los distintos endpoint de la aplicación. La colección se puede exportar en formato Json e incorporar en la aplicación en src/test/postman