**Proyecto Diseño de Software**
**II Semestre 2019**
**Configuración Spring + PostgreSQL**

PostgreSQL es un sistema de administración de bases de datos de propósito general y relacional, además, es el sistema de base de datos de código abierto más avanzado lo que nos conllevo a elegirlo para el desarrollo de nuestro proyecto. En este documento se comentan los requisitos que tenemos que suplir para realizar una conexión entre el Framework Spring y Postgresql.

## Configurar PostgreSQL y la herramienta pgAdmin

Para descargar PostgreSQL, podemos ir al sitio web oficial de PostgreSQL. Seleccionamos el sistema operativo de preferencia y lo descargamos. Luego seguimos las instrucciones que nos brinda el sitio web.
Una vez que instalado, se ejecutará en localhost: 5432 por defecto a menos que cambiaramos el puerto durante la instalación. Ahora, necesitaremos una herramienta cliente para acceder a la base de datos. Para nuestro proyecto utilizaremos PgAdmin.

## Pre requisitos para desarrollar una aplicación con Spring+Postgresql

- PostgreSQL versión 9.5 o superior.
- pgAdmin III o IV
- Gradle
- IDE como IntelliJ en el caso de nuestro proyecto.

## Dependencia de Gradle

Necesitamos configurar las dependencias de Gradle antes de iniciar el proyecto, a continuación se presenta un bloque de código con las dependencias a modo de ejemplo:

```
dependencies {
implementation('org.springframework.boot:spring-boot-starter-data-jpa')
implementation('org.springframework.boot:spring-boot-starter-web')
implementation('org.postgresql:postgresql')
testImplementation('org.springframework.boot:spring-boot-starter-test')
}
```

**Configuración**

Spring-Data utiliza las propiedades spring.datasource para localizar la instancia de postgres y conectarla. Para el ejemplo se a usado spring.jpa.hibernate.ddl-auto = create-drop, pero no es muy común utilizarlo en una aplicación pues lo que hace es limpiar los datos una vez que la aplicación se detiene. También podemos notar la entrada spring.jpa.properties.hibernate.jdbc.lob.non_contextual_creation = true. Esta entrada se coloca solo para evitar un mensaje de advertencia en los registros cuando inicia la aplicación spring-boot. Este error es de hibernate y se presenta cuando intenta recuperar algunos metadatos de postgresql db y no pudo encontrarlos. Sin embargo, no causa ningún problema. Además, debemos asegurarnos de actualizar el nombre de la base de datos en la propiedad spring.datasource.url.A modo de ejemplo tenemos:

```
server.port=9090
spring.jpa.database=POSTGRESQL
spring.datasource.platform=postgres
spring.datasource.url=jdbc:postgresql://localhost:5432/postgres
spring.datasource.username=postgres
spring.datasource.password=root
spring.jpa.show-sql=true
spring.jpa.generate-ddl=true
spring.jpa.hibernate.ddl-auto=create-drop
spring.jpa.properties.hibernate.jdbc.lob.non_contextual_creation=true
```

Con esto ya tenemos Spring configurado para tomar postgreSQL como la base de datos por defecto para nuestro proyecto. El siguiente paso sería investigar los métodos de Spring que controlan la base de datos directamente.

Para resumir, decidimos utilizar la base de datos PostgreSQL porque se está acelerando rápidamente como una opción RDBMS y está obteniendo la ventaja al ser una tecnología de código abierto. Además el framework Spring proporciona una manera fácil de interactuar con PostgreSQL a través de su componente spring data jpa.