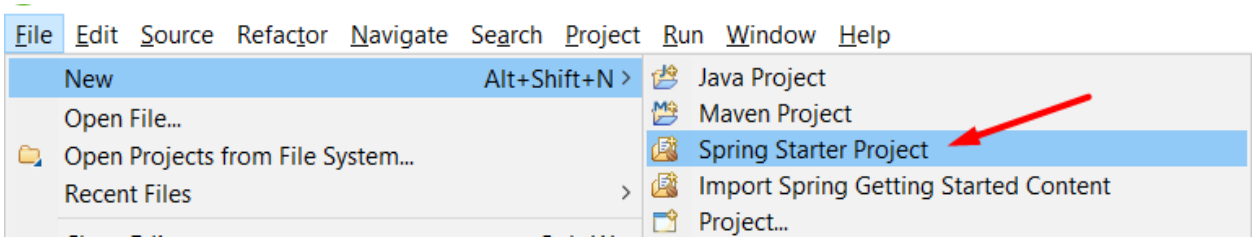
PROYECTO “Generar Pdf”

Paso 1: Crear el Proyecto en STS

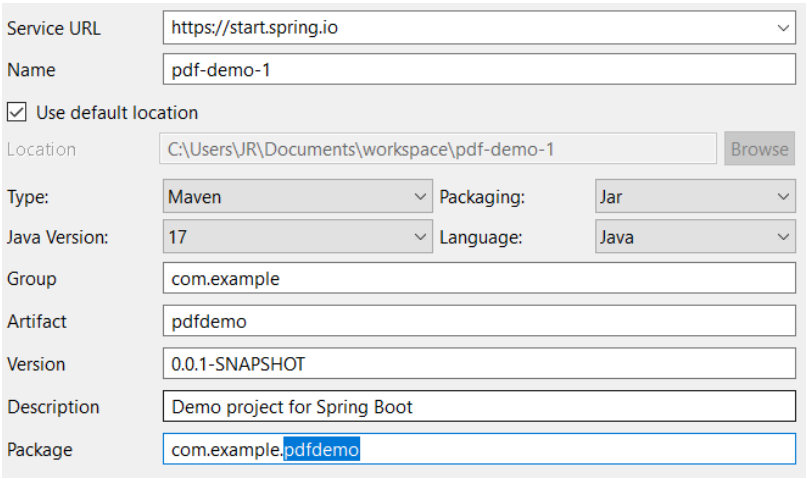
Abrir Spring Tool Suite (STS).

Ir a **File → New → Spring Starter Project**



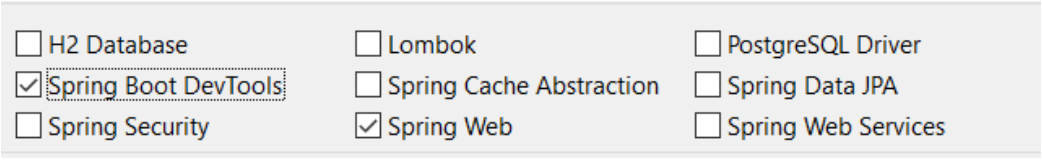
Completar los datos del proyecto:

- **Name:** pdf-demo
- **Type:** Maven
- **Java Version:** 17
- **Group:** com.example
- **Artifact:** pdfdemo
- **Package:** com.example.pdfdemo
- **Packaging:** Jar



Seleccionar las dependencias necesarias:

- **Spring Web**
- **Spring Boot DevTools**



Agregar la dependencia de iText en el pom.xml

```
<!-- iText PDF (versión gratuita para uso educativo) -->
        <dependency>
            <groupId>com.itextpdf</groupId>
            <artifactId>itext7-core</artifactId>
            <version>8.0.1</version>
            <type>pom</type>
        </dependency>
```

Hacer clic en **Finish** para generar el proyecto.

Paso 2: Crear un Servicio para generar el PDF

Creamos la clase service de ReportePDFService

```java
package com.example.pdfdemo.service;

import com.itextpdf.kernel.pdf.PdfWriter;
import com.itextpdf.kernel.pdf.PdfDocument;
import com.itextpdf.layout.Document;
import com.itextpdf.layout.element.Paragraph;
import com.itextpdf.layout.properties.TextAlignment;

import org.springframework.stereotype.Service;

import java.io.ByteArrayOutputStream;

@Service
public class ReportePDFService {

    public byte[] generarPDF() {
        ByteArrayOutputStream outputStream = new ByteArrayOutputStream();

        PdfWriter writer = new PdfWriter(outputStream);
        PdfDocument pdf = new PdfDocument(writer);
        Document document = new Document(pdf);

        document.add(new Paragraph("Reporte PDF generado con iText")
                .setBold()
                .setFontSize(16)
                .setTextAlignment(TextAlignment.CENTER));

        document.add(new Paragraph("Este es un reporte generado en tiempo real desde una API REST.")
                .setFontSize(12));

        document.add(new Paragraph("Fecha: " + java.time.LocalDate.now()));

        document.close();
        return outputStream.toByteArray();
    }
}
```

Explicación:

- **@Service**: Esta anotación indica que la clase es un servicio de Spring.
- **ByteArrayOutputStream outputStream = new ByteArrayOutputStream();**: Crea un flujo de salida en memoria que almacenará los bytes del PDF generado.
- **PdfWriter writer = new PdfWriter(outputStream);**: Crea un objeto PdfWriter que escribirá el contenido del PDF en el ByteArrayOutputStream.
- **PdfDocument pdf = new PdfDocument(writer);**: Crea un documento PDF utilizando el PdfWriter.

- **Document document = new Document(pdf);**: Crea un objeto Document que permite agregar contenido al PDF, como párrafos, imágenes, tablas, etc.
- **document.add(new Paragraph("Reporte PDF generado con iText")...)**: Agrega un párrafo al documento con el texto "Reporte PDF generado con iText". El párrafo se configura con un tamaño de fuente de 16, en negrita y centrado.
- **document.add(new Paragraph("Este es un reporte generado en tiempo real desde una API REST."))**: Agrega otro párrafo con un tamaño de fuente de 12.
- **document.add(new Paragraph("Fecha: " + java.time.LocalDate.now()));**: Agrega un párrafo con la fecha actual.
- **document.close();**: Cierra el documento, lo que finaliza la creación del PDF.
- **return outputStream.toByteArray();**: Devuelve el contenido del PDF como un arreglo de bytes, que luego se envía como respuesta en el controlador.

Paso 3: Crear el Controlador REST

Creamos la clase controlador ReporteController

```
package com.example.pdfdemo.controller;

import com.example.pdfdemo.service.ReportePDFService;
import org.springframework.http.HttpHeaders;
import org.springframework.http.MediaType;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

@RestController
@RequestMapping("/pdf")
public class ReporteController {

    private final ReportePDFService reportePDFService;

    public ReporteController(ReportePDFService reportePDFService) {
        this.reportePDFService = reportePDFService;
    }

    @GetMapping("/generar")
    public ResponseEntity<byte[]> generarReportePDF() {
        byte[] pdfBytes = reportePDFService.generarPDF();

        return ResponseEntity.ok()
                .header(HttpHeaders.CONTENT_DISPOSITION, "attachment; filename=reporte.pdf")
                .contentType(MediaType.APPLICATION_PDF)
                .body(pdfBytes);
    }
}
```

Explicación:

- **@RestController**: Esta anotación indica que la clase es un controlador REST, lo que significa que maneja solicitudes HTTP y devuelve respuestas en formato JSON, XML, o en este caso, un archivo PDF.
- **@RequestMapping("/pdf")**: Define la ruta base para todas las solicitudes manejadas por este controlador. En este caso, todas las URLs que comiencen con /pdf serán manejadas por este controlador.
- **ReportePDFService reportePDFService**: Es una dependencia inyectada en el controlador a través del constructor. Este servicio se encarga de generar el PDF.
- **@GetMapping("/generar")**: Este método maneja las solicitudes GET a la ruta /pdf/generar. Cuando un cliente hace una solicitud GET a esta URL, se ejecuta este método.
- **ResponseEntity<byte[]>**: Es un objeto que representa la respuesta HTTP. En este caso, la respuesta contiene un arreglo de bytes (byte[]) que representa el contenido del PDF.
- **reportePDFService.generarPDF()**: Llama al método generarPDF() del servicio para generar el PDF y obtenerlo como un arreglo de bytes.
- **ResponseEntity.ok()**: Crea una respuesta HTTP con el código de estado 200 (OK).
- **.header(HttpHeaders.CONTENT_DISPOSITION, "attachment; filename=reporte.pdf")**: Agrega un encabezado a la respuesta HTTP que indica que el contenido debe ser descargado como un archivo adjunto con el nombre reporte.pdf.
- **.contentType(MediaType.APPLICATION_PDF)**: Establece el tipo de contenido de la respuesta como application/pdf, lo que indica que el contenido es un archivo PDF.
- **.body(pdfBytes)**: Establece el cuerpo de la respuesta con el contenido del PDF en forma de arreglo de bytes.

Paso 4: Ejecutar los endpoints en Postman

GET http://localhost:8080/pdf/generar