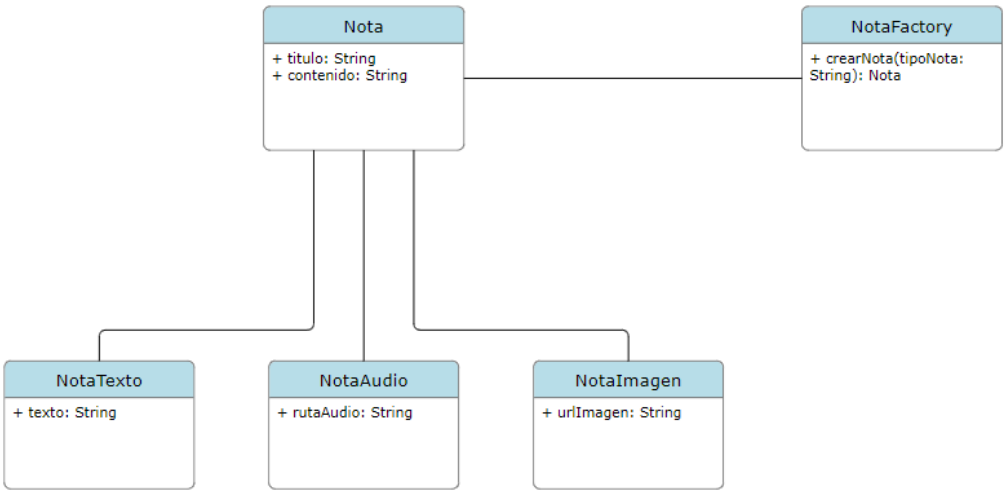
Bloque II – S7 Patrones de Diseño: Factory Method

1. Diagrama de clases



2. Código de las clases (Sin entrar en detalles)

```
// Clase abstracta que representa una nota.
abstract class Nota {
    // Título de la nota.
    var titulo: String = ""

    // Contenido de la nota.
    var contenido: String = ""

    // Devuelve el título de la nota.
    fun getTitulo(): String {
        return this.titulo
    }

    // Establece el título de la nota.
    fun setTitulo(titulo: String) {
        this.titulo = titulo
    }

    // Devuelve el contenido de la nota.
    fun getContenido(): String {
        return this.contenido
    }

    // Establece el contenido de la nota.
    fun setContenido(contenido: String) {
        this.contenido = contenido
    }
}

// Clase que representa una nota de texto.
class NotaTexto : Nota() {
    // Texto de la nota.
    var texto: String = ""

    // Devuelve el texto de la nota.
    fun getTexto(): String {
        return this.texto
    }

    // Establece el texto de la nota.
    fun setTexto(texto: String) {
        this.texto = texto
    }
}
```

```
// Clase que representa una nota de imagen.
class NotaImagen : Nota() {
    // URL de la imagen de la nota.
    var urlImagen: String = ""

    // Devuelve la URL de la imagen de la nota.
    fun getUrlImagen(): String {
        return this.urlImagen
    }

    // Establece la URL de la imagen de la nota.
    fun setUrlImagen(urlImagen: String) {
        this.urlImagen = urlImagen
    }
}

// Clase que representa una nota de audio.
class NotaAudio : Nota() {
    // Ruta del archivo de audio de la nota.
    var rutaAudio: String = ""

    // Devuelve la ruta del archivo de audio de la nota.
    fun getRutaAudio(): String {
        return this.rutaAudio
    }

    // Establece la ruta del archivo de audio de la nota.
    fun setRutaAudio(rutaAudio: String) {
        this.rutaAudio = rutaAudio
    }
}

abstract class NotaFactory {
    // Devuelve un objeto de tipo Nota del tipo especificado.
    abstract fun crearNota(tipoNota: String): Nota
}
```

3. Reflexiona sobre cambios futuros

a. ¿Qué pasa si en un futuro se quisiera añadir un nuevo tipo de notas?

   Si en un futuro se quisiera añadir un nuevo tipo de notas, solo sería necesario crear una nueva clase que herede de la clase Nota y que defina los atributos específicos del nuevo tipo de nota.

b. ¿Qué partes de tu aplicación tendrías que modificar?

   Las partes de la aplicación que tendríamos que modificar son:
   La clase NotaFactory para agregar un nuevo método de creación de notas del nuevo tipo.

c. ¿Qué nuevas clases tendrías que añadir?

   Una clase que represente el nuevo tipo de nota. En nuestro ejemplo, la clase NotaVideo.

   Una clase que implemente el método de creación de notas del nuevo tipo en la clase NotaFactory. En nuestro ejemplo, la clase NotaVideoFactory que extiende de la clase NotaFactory y que implementa el método crearNota() para crear notas de vídeo.