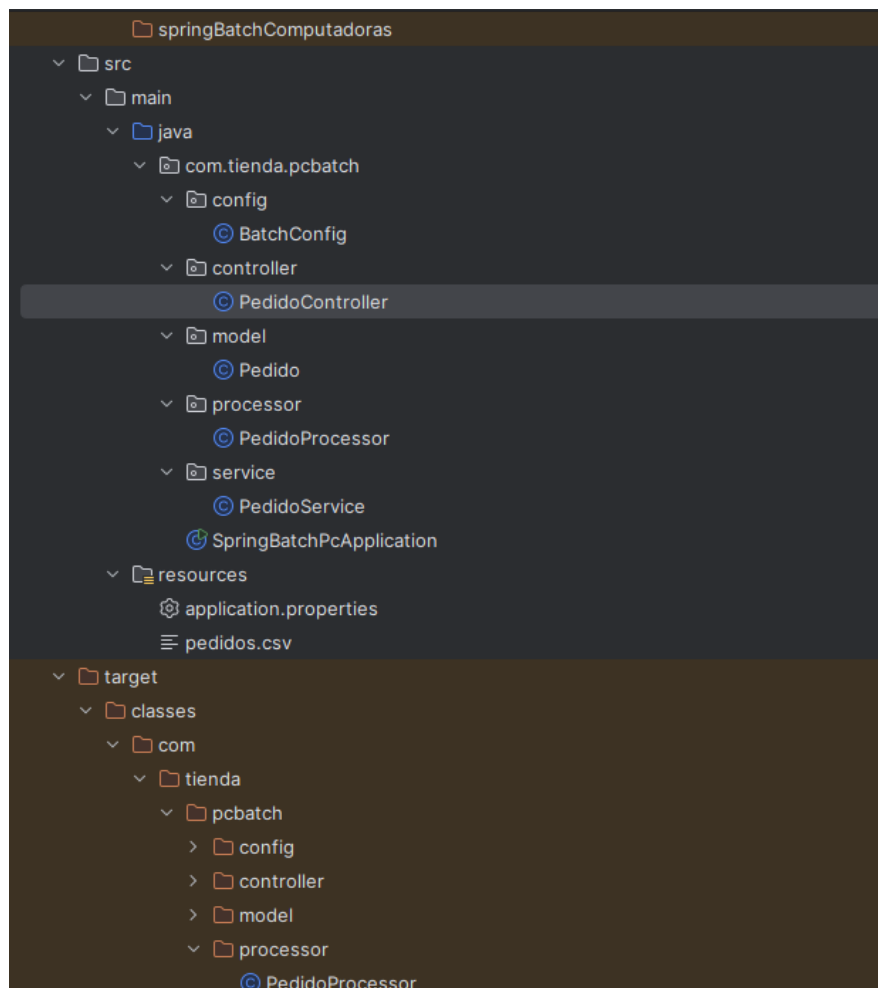


Proyecto SprintBatch con Spring Boot

Una visualización de la arquitectura para el procesamiento de archivos CSV.

Este proyecto implementa un proceso batch usando Spring Boot y Spring Batch, enfocado en la carga y procesamiento de pedidos desde un archivo CSV hacia una base de datos MySQL. Además, se incluye un servicio REST para la subida de archivos CSV mediante un endpoint `/api/pedidos`

Estructura del Proyecto



El flujo principal consiste en:

1. Lectura del archivo CSV.
2. Procesamiento de cada registro (por ejemplo, transformaciones o validaciones).
3. Escritura de los datos en la tabla pedidos de MySQL.
4. Manejo de errores y validación de integridad de dato

Arquitectura de Alto Nivel

El sistema está diseñado en capas, garantizando la separación de responsabilidades y la escalabilidad. El flujo de datos es claro y eficiente, desde la subida del archivo por el usuario hasta su persistencia final en la base de datos.



3. Configuración de Spring Batch (BatchConfig.java)

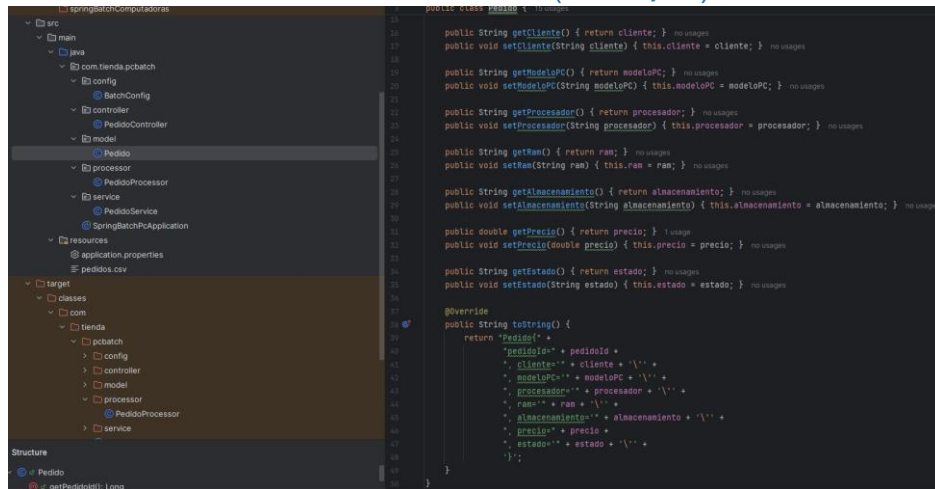
El método principal y recomendado. Procesa datos en lotes transaccionales (`chunks`), ofreciendo tolerancia a fallos, reinicio automático y un rendimiento óptimo para grandes volúmenes de datos.

3.1 Flujo 1: Proceso Robusto con Spring Batch



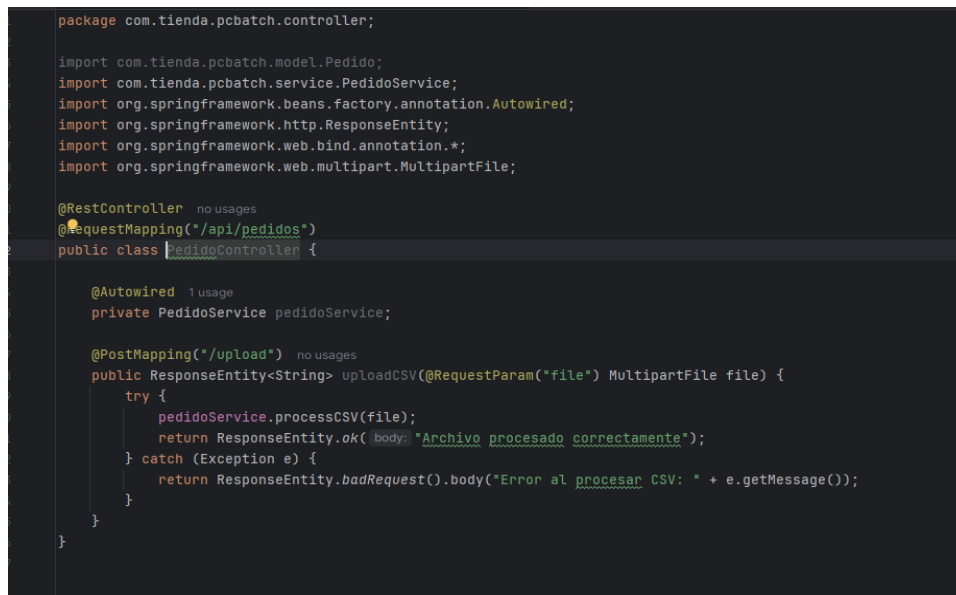
- ✓ FlatFileItemReader permite leer CSV automáticamente.
- ✓ chunk(10) procesa 10 registros por transacción.
- ✓ JdbcBatchItemWriter realiza la inserción en la base de datos usando JDBC.
- ✓ PedidoProcessor puede contener lógica de validación o transformación.

4. Modelo de Datos (Pedido.java)

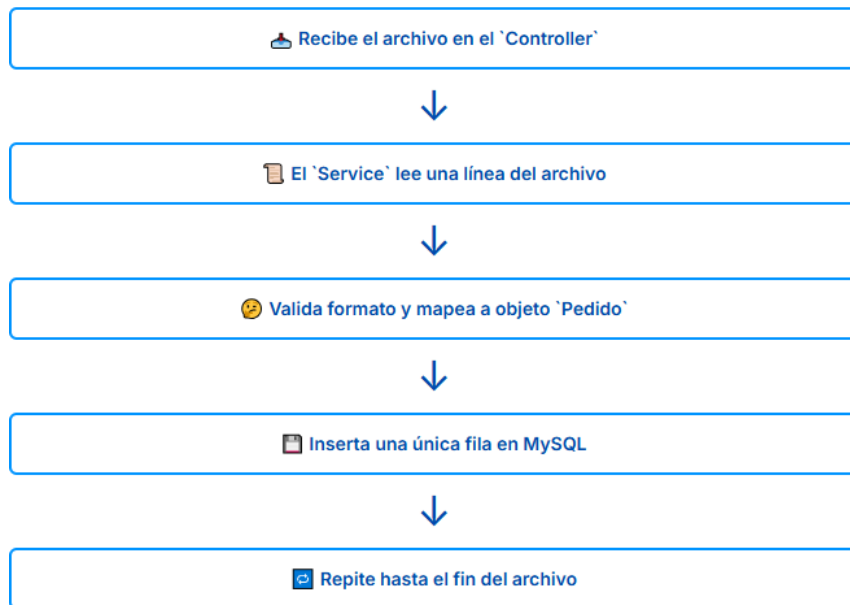


Controlador REST (PedidoController.java)

- ✓ El servicio procesa el archivo línea por línea.
- ✓ Endpoint POST /api/pedidos/upload recibe un archivo CSV.
- ✓ Llama al servicio PedidoService para procesar e insertar los datos.



Flujo 2: Carga Manual vía Servicio REST



- ✓ Inserta cada registro en la base de datos usando JdbcTemplate.
- ✓ Se sube un archivo CSV en properties y se hace una consulta con el post para poder ver los registros hechos /api/pedidos
- ✓ PedidoService procesa cada registro:
- ✓ Valida número de columnas.
- ✓ Inserta datos en MySQL.

Pruebas y Resultados

La imagen muestra una interfaz de desarrollo con el código de la clase `PedidoProcessor` y los logs de ejecución de la aplicación `SpringBatchPcApplication`.

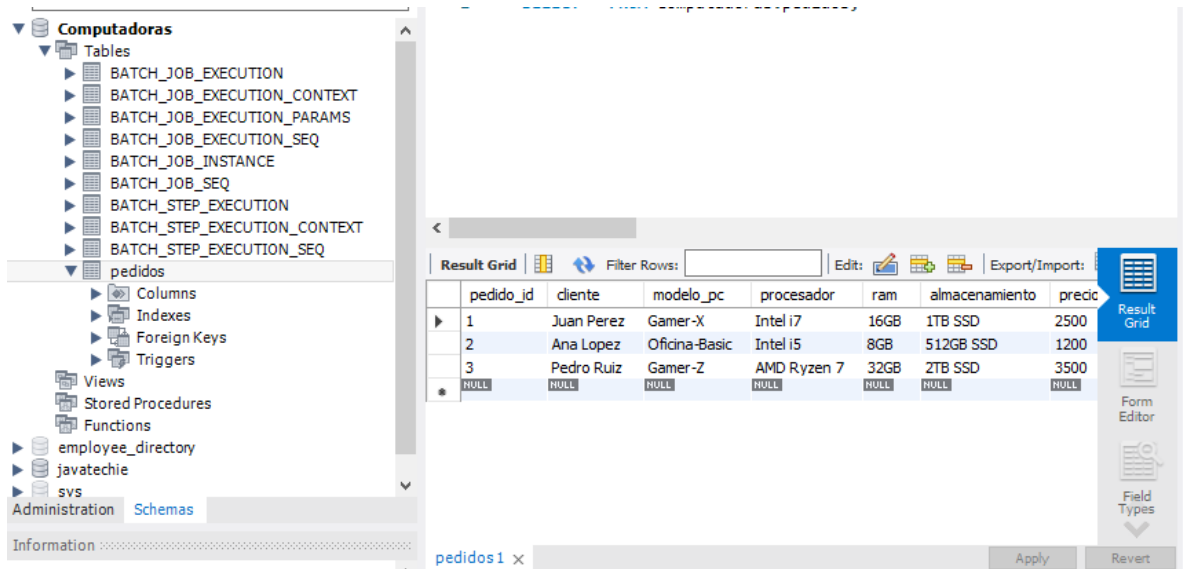
Código de la clase `PedidoProcessor`:

```
24  
25 @Override  
26 public void run(String... args) throws Exception {  
27     JobParameters params = new JobParametersBuilder()  
28         .addLong("key", "time", System.currentTimeMillis())  
29         .toJobParameters();  
30     jobLauncher.run(importPedidoJob, params);  
31 }  
32  
33
```

Logs de ejecución:

```
2025-09-19 17:21:01.944 INFO 4360 --- [localhost:27017] org.mongodb.driver.connection : Opened connection [connectionId{localValue:2, serverValue:44}] to localhost:27017  
2025-09-19 17:21:01.944 INFO 4360 --- [localhost:27017] org.mongodb.driver.connection : Opened connection [connectionId{localValue:1, serverValue:43}] to localhost:27017  
2025-09-19 17:21:01.944 INFO 4360 --- [localhost:27017] org.mongodb.driver.cluster : Monitor thread successfully connected to server with description ServerDescription{address=localhost:27017, type=STANDALONE, state=CONNECTED, topologyVersion=TopologyVersion{majorVersion=0, minorVersion=0}, serverCapabilities=ServerCapabilities{maxWireVersion=13, minWireVersion=0}}  
2025-09-19 17:21:02.272 INFO 4360 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 9191 (http) with context path ''  
2025-09-19 17:21:02.494 INFO 4360 --- [main] c.t.pcbatch.SpringBatchPcApplication : Started SpringBatchPcApplication in 5.479 seconds (JVM running for 6.006)  
2025-09-19 17:21:02.494 INFO 4360 --- [main] o.s.b.c.l.support.SimpleJobLauncher : Job: [FlowJob: [name=importPedidoJob]] launched with the following parameters: [{time=1758324062276}]  
2025-09-19 17:21:02.593 INFO 4360 --- [main] o.s.batch.core.job.SimpleStepHandler : Executing step: [step1]  
Pedido rechazado (precio inválido): Pedido(pedidoId=4, cliente='Luis Mendez', modeloPc='Oficina-Basic', procesador='Intel i3', ram='4GB', almacenamiento='256GB SSD', precio=0.0, estado='PENDIENTE')  
2025-09-19 17:21:02.707 INFO 4360 --- [main] o.s.batch.core.step.AbstractStep : Step: [step1] executed in 113ms  
2025-09-19 17:21:02.759 INFO 4360 --- [main] o.s.b.c.l.support.SimpleJobLauncher : Job: [FlowJob: [name=importPedidoJob]] completed with the following parameters: [{time=1758324062276}] and the following
```

Efectivamente está corriendo sin ningún problema



Configuración de Properties

```
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url = jdbc:mysql://localhost:3306/Computadoras
spring.datasource.username = root
spring.datasource.password = xideral234
spring.jpa.show-sql = true
spring.jpa.hibernate.ddl-auto = update
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5Dialect
server.port=9191
spring.batch.initialize-schema=ALWAYS

#disabled job run at startup
spring.batch.job.enabled=false
```