==============

Spring Batch

==============

=> Spring Batch is a lightweight, comprehensive batch framework designed to enable the development of robust batch applications.

============================

What is Batch Application ?

============================

-> The application which will process bulk of records is called as Batch application.

Ex :

1) Sending Monthly Salaries to all employees in company

2) Generating Monthly Payslips for employees

3) Sending Monthly Bank Account statemet to acc holders

4) Sending Monthly credit card bill statement to card holders

5) Sending Festival Greetings to all customers

6) Sending Notices on daily/weekly/monthly/yearly basis...

```
                              process                                          process
CSV File  ==============> Database  ================>   CSV File
```

=========================

Spring Batch Terminology

=========================

1) JobLauncher

2) Job

3) Step

4) ItemReader

5) ItemProcessor

6) ItemWriter

7) JobRepository

===============================

Spring Boot with Batch Example

===============================

Requirement : Read Customers Data From CSV file and write into Database table

1) Create Spring Starter project with below dependencies

    a) web-starter

    b) batch

c) data-jpa

d) mysql-driver

e) lombok

2) Configure Datasource properites in application.yml file

```
spring.application.name=Batch-App
spring.datasource.username=root
spring.datasource.password=password
spring.datasource.url=jdbc:mysql://localhost:3306/batchdemo
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
```

3) Keep Source File under src/main/resources folder (customers.csv)

Taken from github link here : https://github.com/ashokitschool/sringboot_batch_app.git

4) Create Entity class & Repository interface

```
package in.api.entity;

import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.Id;
import jakarta.persistence.Table;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;


@Entity
@Table(name="CUSTOMERS_INFO")
@Data
@AllArgsConstructor
@NoArgsConstructor
public class Customer {

	@Id
	@Column(name = "CUSTOMER_ID")
	private int id;

	@Column(name = "FIRST_NAME")
	private String firstName;

	@Column(name = "LAST_NAME")
	private String lastName;

	@Column(name = "EMAIL")
	private String email;

	@Column(name = "GENDER")
	private String gender;

	@Column(name = "CONTACT")
	private String contactNo;
```

```
        @Column(name = "COUNTRY")

        private String country;


        @Column(name = "DOB")

        private String dob;


    }



    package in.api.repo;


    import java.io.Serializable;


    import org.springframework.data.jpa.repository.JpaRepository;


    import in.api.entity.Customer;


    public interface CustomerRepository extends
JpaRepository<Customer,Serializable>{


    }
```

5) Create Batch Configuration class

```
    package in.api.config;
```

```
import org.springframework.batch.core.Job;
import org.springframework.batch.core.Step;
import
org.springframework.batch.core.configuration.annotation.EnableBatchProcessing;
//import
org.springframework.batch.core.configuration.annotation.JobBuilderFactory;
import org.springframework.batch.core.job.builder.JobBuilder;
import org.springframework.batch.core.repository.JobRepository;
import org.springframework.batch.core.step.builder.StepBuilder;
import org.springframework.batch.item.data.RepositoryItemWriter;
import org.springframework.batch.item.file.FlatFileItemReader;
import org.springframework.batch.item.file.LineMapper;
import org.springframework.batch.item.file.mapping.BeanWrapperFieldSetMapper;
import org.springframework.batch.item.file.mapping.DefaultLineMapper;
import org.springframework.batch.item.file.transform.DelimitedLineTokenizer;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.core.io.FileSystemResource;
import org.springframework.core.task.SimpleAsyncTaskExecutor;
import org.springframework.core.task.TaskExecutor;
import org.springframework.transaction.PlatformTransactionManager;

import in.api.entity.Customer;
import in.api.repo.CustomerRepository;

@Configuration
//@EnableBatchProcessing
```

```
          public class CsvBatchConfig {


                    @Autowired

                    private CustomerRepository repo;


                    //create Reader

                    @Bean

                    public FlatFileItemReader<Customer> customerReader(){

                              FlatFileItemReader<Customer> itemReader = new
FlatFileItemReader<>();

                              itemReader.setResource(new
FileSystemResource("src/main/resources/customers.csv"));

                              itemReader.setName("csv-reader");

                              itemReader.setLinesToSkip(1);

                              itemReader.setLineMapper(lineMapper());




                              return itemReader;

                    }


                    private LineMapper<Customer> lineMapper(){

                              DefaultLineMapper<Customer> lineMapper= new
DefaultLineMapper<>();

                              DelimitedLineTokenizer lineTokenizer = new DelimitedLineTokenizer();

                              lineTokenizer.setDelimiter(",");

                              lineTokenizer.setStrict(false);


     lineTokenizer.setNames("id","firstName","lastName","email","gender","contactNo","country","
dob");
```

```
            BeanWrapperFieldSetMapper<Customer> fieldSetMapper = new
BeanWrapperFieldSetMapper<>(); // convert csv data into java object
            fieldSetMapper.setTargetType(Customer.class);


            lineMapper.setLineTokenizer(lineTokenizer);
            lineMapper.setFieldSetMapper(fieldSetMapper);
            return lineMapper;
        }


        //create Processor
        @Bean
        public CustomerProcessor customerProcessor() {
            return new CustomerProcessor();
        }


        //create Writer
        @Bean
        public RepositoryItemWriter<Customer> customerWriter(){
            RepositoryItemWriter<Customer> repositoryItemWriter = new
RepositoryItemWriter<>();
            repositoryItemWriter.setRepository(repo);
            repositoryItemWriter.setMethodName("save");


            return repositoryItemWriter;


        }


        //create Step
```

```
                    @Bean
                    public Step customerStep(JobRepository jobRepository,
PlatformTransactionManager transactionManager) {
                        return new StepBuilder("customerStep",jobRepository)
                                .<Customer, Customer>chunk(10,
transactionManager) // Process 10 records at a time
                                .reader(customerReader())
                                .processor(customerProcessor())
                                .writer(customerWriter())
                                .taskExecutor(taskExecutor())
                                .build();



                    }


                    @Bean
                    public TaskExecutor taskExecutor() {
                        return new SimpleAsyncTaskExecutor();
                    }



                    //create Job


                    @Bean
                    public Job job( JobRepository jobRepository, PlatformTransactionManager
transactionManager) {
                        return new JobBuilder("myjob", jobRepository)

    .flow(customerStep(jobRepository,transactionManager))
                                .end()
                        .build();
        }

}

package in.api.config;

import org.springframework.batch.item.ItemProcessor;

import in.api.entity.Customer;

public class CustomerProcessor implements
ItemProcessor<Customer,Customer>{

        @Override
        public Customer process(Customer item) throws Exception {

                return item;
        }
}
```

6) Create RestController and launch job

```
package in.api.rest;

import org.springframework.batch.core.Job;
import org.springframework.batch.core.JobParameters;
import org.springframework.batch.core.JobParametersBuilder;
import org.springframework.batch.core.launch.JobLauncher;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class CustomerRestController {

    @Autowired
    private JobLauncher jobLauncher;

    @Autowired
    private Job job;

    @GetMapping("/customers")
    public void loadCsvToDb() throws Exception {
        JobParameters jobParams =
                new JobParametersBuilder().addLong("Start-
At",System.currentTimeMillis()).toJobParameters();

        jobLauncher.run(job,jobParams);
    }


}
```

Hit below URL in Postman

Method GET --> http://localhost:8080/customers

Git Hub Repo URL : https://github.com/ashokitschool/sringboot_batch_app.git