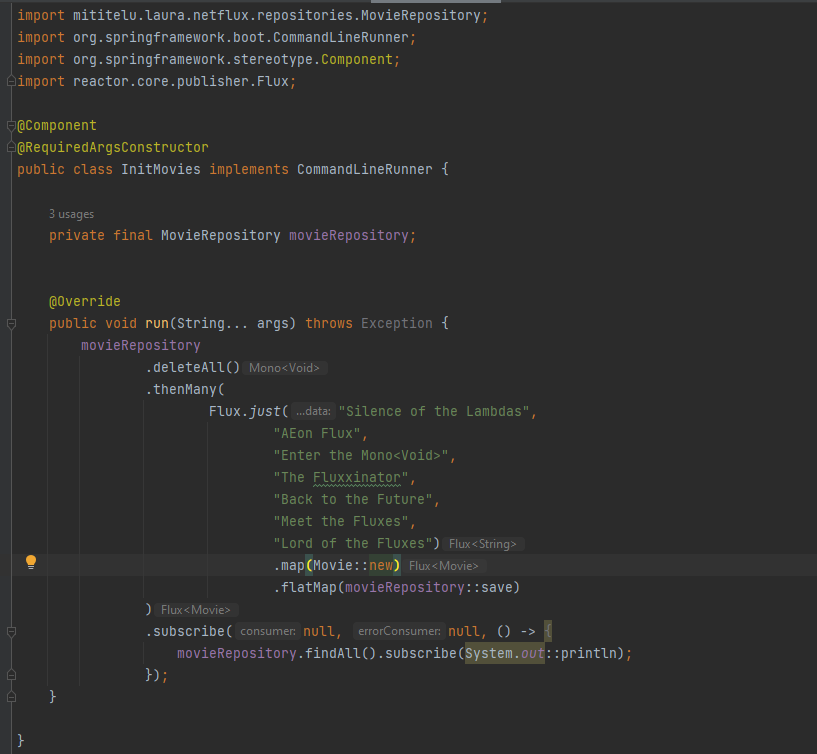
Manually added dependency for mongo db

<dependency>  
 <groupId>de.flapdoodle.embed</groupId>  
 <artifactId>de.flapdoodle.embed.mongo.spring30x</artifactId>  
 <version>4.11.0</version>  
<!-- <scope>test</scope>-->  
 </dependency>

Commented scope test!

@Document  
@Data  
@NoArgsConstructor  
@RequiredArgsConstructor  
public class Movie {  
  
 private String id;  
  
 @NonNull  
 private String title;  
}

@Repository  
public interface MovieRepository extends ReactiveMongoRepository<Movie, String> {  
}



<https://www.garretwilson.com/blog/2023/04/02/embedded-mongodb-ide-spring-boot-3>

RestController automatically provides for a response body

Cmd

curl -v <http://localhost:8080/movies>

[{"id":"669927f2f5465d25ec2a9563","title":"Silence of the Lambdas"},{"id":"669927f2f5465d25ec2a9565","title":"Enter the Mono<Void>"},{"id":"669927f2f5465d25ec2a9564","title":"AEon Flux"},{"id":"669927f2f5465d25ec2a9566","title":"The Fluxxinator"},{"id":"669927f2f5465d25ec2a9567","title":"Back to the Future"},{"id":"669927f2f5465d25ec2a9569","title":"Lord of the Fluxes"},{"id":"669927f2f5465d25ec2a9568","title":"Meet the Fluxes"}]\* Connection #0 to host localhost left intact

curl -v <http://localhost:8080/movies/669927f2f5465d25ec2a9564>

\* Host localhost:8080 was resolved.

\* IPv6: ::1

\* IPv4: 127.0.0.1

\* Trying [::1]:8080...

\* Connected to localhost (::1) port 8080

> GET /movies/669927f2f5465d25ec2a9564 HTTP/1.1

> Host: localhost:8080

> User-Agent: curl/8.7.1

> Accept: \*/\*

>

\* Request completely sent off

< HTTP/1.1 200 OK

< Content-Type: application/json

< Content-Length: 53

<

{"id":"669927f2f5465d25ec2a9564","title":"AEon Flux"}\* Connection #0 to host localhost left intact

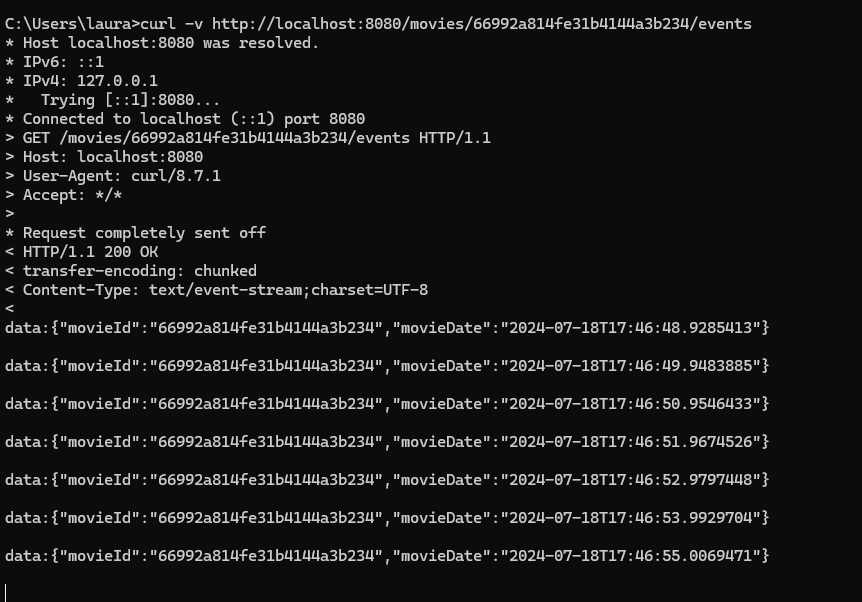
Streams

@Override  
public Flux<MovieEvent> streamMovieEvents(String id) {  
 return Flux.<MovieEvent>*generate*(movieEventSynchronousSink -> {  
 movieEventSynchronousSink.next(new MovieEvent(id, LocalDateTime.*now*()));  
 }).delayElements(Duration.*ofSeconds*(1)); //will emit a new movie event every second!  
}

@GetMapping(value = "{id}/events", produces = MediaType.*TEXT\_EVENT\_STREAM\_VALUE*)  
Flux<MovieEvent> streamMovieEvents(@PathVariable String id){  
 return movieService.streamMovieEvents(id);  
}

produces = MediaType.*TEXT\_EVENT\_STREAM\_VALUE*

it is important to add so that it gets encoded properly!!



Use a client that supports Streaming.

?Postman might not be able to use it