

Subject	SWA – CS590
Assignment	Lab 11
Date	9/11/2021
Students	Vo Ngoc Huy Nguyen - 611066
Professor	Rene de Jong

Lab Assignment 11: Responsive and Distributed Transaction

Exercise 1: Spring WebFlux

REST stock service and client

workspace-spring-tool-suite-4-4.11.0.RELEASE_2 - StockService/src/main/java/com/StockServiceApplication.java - Spring Tool Suite 4

```

1 package com;
2
3
4 import org.springframework.boot.SpringApplication;
5 import org.springframework.boot.autoconfigure.SpringBootApplication;
6 import org.springframework.context.annotation.ComponentScan;
7
8
9
10 @ComponentScan("com")
11 @SpringBootApplication
12
13 public class StockServiceApplication {
14
15     public static void main(String[] args) {
16         SpringApplication.run(StockServiceApplication.class, args);
17     }
18
19
20 }
21

```

workspace-spring-tool-suite-4-4.11.0.RELEASE_2 - StockClient/src/main/java/com/WelcomeTask.java - Spring Tool Suite 4

```

1 package com;
2
3 import java.time.LocalDateTime;
4
5 import org.springframework.scheduling.annotation.Scheduled;
6 import org.springframework.stereotype.Component;
7 import org.springframework.web.reactive.function.client.WebClient;
8
9 import com.domain.Stock;
10
11 import reactor.core.publisher.Flux;
12
13 @Component
14 public class WelcomeTask {
15
16     @Scheduled(fixedRate = 5000)
17     public void welcome() {
18         Flux<Stock> result = WebClient.create("http://localhost:8080/stock").get().retrieve().bodyToFlux(Stock.class);
19         result.subscribe(s -> {
20             System.out.print(LocalDateTime.now() + " : ");
21             System.out.println(s);
22         });
23     }
24 }
25

```

Refer the output:

The screenshot shows an IDE window titled "workspace-spring-tool-suite-4-4.11.0.RELEASE_2 - StockClient/src/main/java/com/StockClientApplication.java - Spring Tool Suite 4". The main editor displays the source code for `StockClientApplication.java`. The code imports various Spring and Reactor dependencies and implements a `main` method that uses `WebClient` to fetch stock data from `http://localhost:8080/stock` and prints it to the console. The Package Explorer on the left shows the project structure, including `com` and `src/main/java` packages. The Console window at the bottom shows the output of the application, displaying a series of stock prices for MSFT over time.

```
import org.springframework.http.converter.json.MappingJackson2HttpMessageConverter;
import org.springframework.scheduling.annotation.EnableScheduling;
import org.springframework.web.client.RestTemplate;
import org.springframework.web.reactive.function.client.WebClient;
import com.domain.Stock;
import reactor.core.publisher.Flux;
import java.time.LocalDateTime;

@SpringBootApplication
@EnableScheduling
public class StockClientApplication {

    public static void main(String[] args) throws InterruptedException {
        SpringApplication.run(StockClientApplication.class, args);
    }

    /*
    Flux<Stock> result = WebClient.create("http://localhost:8080/stock").get().retrieve()
        .bodyToFlux(Stock.class);
    result.subscribe(s -> {
        System.out.print(LocalDateTime.now() + " : ";
        System.out.println(s);
    });
    */
}
```

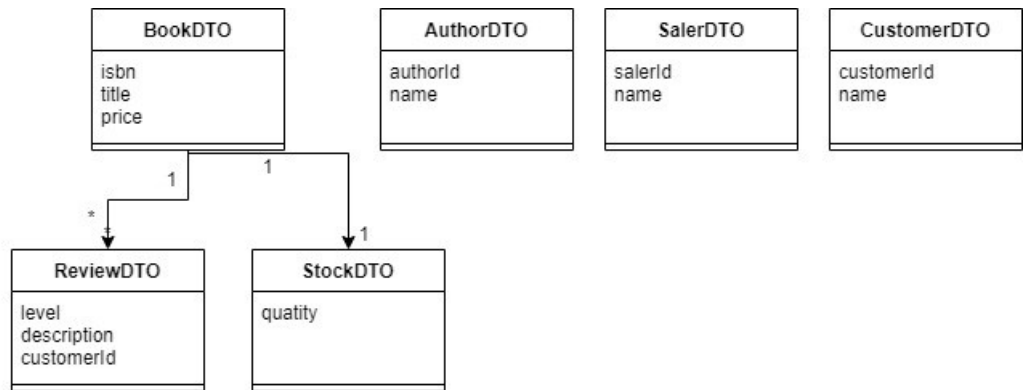
Console Output:

```
2021-09-11T21:13:56.192187400 : stock MSFT has value 278.17
2021-09-11T21:14:00.793529900 : stock MSFT has value 290.1
2021-09-11T21:14:00.808044100 : stock MSFT has value 278.17
2021-09-11T21:14:01.194700600 : stock MSFT has value 291.21
2021-09-11T21:14:05.789200300 : stock MSFT has value 290.1
2021-09-11T21:14:05.805552900 : stock MSFT has value 278.17
2021-09-11T21:14:05.820513900 : stock MSFT has value 291.21
2021-09-11T21:14:06.196957300 : stock MSFT has value 300.45
2021-09-11T21:14:10.789104300 : stock MSFT has value 290.1
2021-09-11T21:14:10.803057 : stock MSFT has value 278.17
2021-09-11T21:14:10.819004100 : stock MSFT has value 291.21
2021-09-11T21:14:10.834323400 : stock MSFT has value 300.45
2021-09-11T21:14:11.208283800 : stock MSFT has value 288.89
```

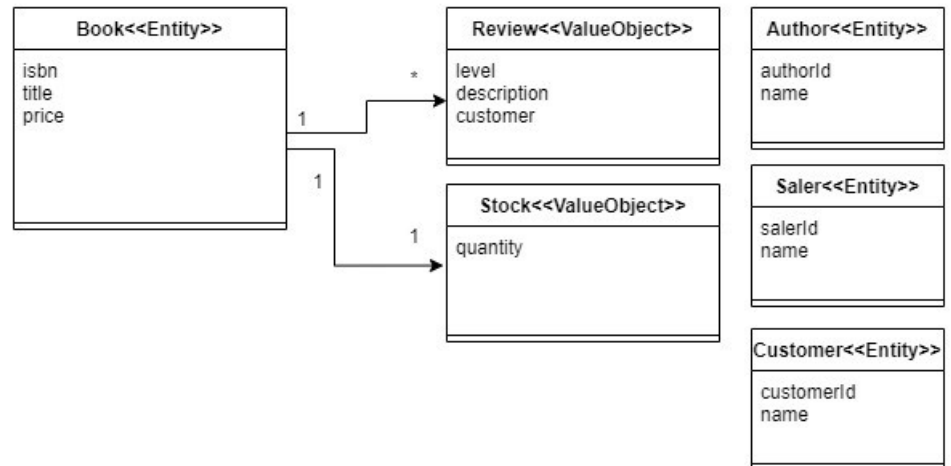
Exercise 2:

Design and draw diagram with BookCatalog service (query and command services)

BookCatalogRead Service
findBookByTitle(title) findBookByAuthor(author) getBook(isbn) findReview(review) getStock(isbn)



BookCommandService
setStock(isbn, quantity) addBook(BookDTO) removeBook(isbn) addReview(isbn, reviewDTO) addSaler(isbn, salerId) addAuthor(isbn, authorId)



Exercise 3:

Draw diagram for webshop

