**Application.java**

**package** com.cognizant;

**import** org.springframework.context.ApplicationContext;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

/\*\*

\* **@author** Saugata Ray

\*

\*/

**public** **class** Application {

**public** **static** **void** main(String[] args) {

ApplicationContext applicationContext=**new** ClassPathXmlApplicationContext("applicationContext.xml");

CommandLineRunner commandLineRunner = (CommandLineRunner) applicationContext.getBean("commandLineRunner");

commandLineRunner.run();

}

}

**CommandLineRunner.java**

**package** com.cognizant;

**import** java.io.BufferedReader;

**import** java.io.IOException;

**import** java.io.InputStreamReader;

**import** java.time.LocalDate;

**import** com.cognizant.model.Book;

**import** com.cognizant.model.Subject;

**import** com.cognizant.service.BookService;

**import** com.cognizant.service.SubjectService;

/\*\*

\* **@author** Saugata Ray

\*

\*/

**public** **class** CommandLineRunner {

**private** SubjectService subjectService;

**private** BookService bookService;

**public** **void** run() {

**while** (**true**) {

System.***out***.println("\n");

System.***out***.println("enter '1' to add a new subject");

System.***out***.println("enter '2' to add a new book");

System.***out***.println("enter '3' to delete a subject");

System.***out***.println("enter '4' to delete a book");

System.***out***.println("enter '5' to search a subject");

System.***out***.println("enter '6' to search a book");

System.***out***.println("enter '7' to show all");

System.***out***.println("enter '8' to exit");

String input = gatValFromConsole("Enter your choise ");

select(Integer.*parseInt*(input));

}

}

**public** **void** select(**int** i) {

**switch** (i) {

**case** 1:

addSubject();

**break**;

**case** 2:

addBook();

**break**;

**case** 3:

deleteSubject();

**break**;

**case** 4:

deleteBook();

**break**;

**case** 5:

searchSubject();

**break**;

**case** 6:

searchBook();

**break**;

**case** 7:

showAll();

**break**;

**case** 8:

exit();

**break**;

}

}

**private** **void** addBook() {

Book book = **new** Book();

book.setBookId(Long.*parseLong*(gatValFromConsole("enter book id")));

book.setTitle(gatValFromConsole("enter book title"));

book.setPrice(Double.*parseDouble*(gatValFromConsole("enter book price")));

book.setVolume(Integer.*parseInt*(gatValFromConsole("enter book volume")));

book.setPublishDate(LocalDate.*parse*(gatValFromConsole("enter book publish date(yyyy-mm-dd) ")));

Book bookObj = bookService.addBook(book);

**if** (**null** != bookObj) {

System.***out***.println("below mentioned book has been added successfully");

System.***out***.println(book.toString());

} **else** {

System.***out***.println("unable to add book");

}

}

**private** **void** deleteBook() {

**long** bookId = Long.*parseLong*(gatValFromConsole("enter the book id to be deleted"));

**if** (bookService.deleteBook(bookId))

System.***out***.println("book with id :" + bookId + " has been removed");

}

**private** **void** searchBook() {

**long** bookId = Long.*parseLong*(gatValFromConsole("enter the book id for search"));

Book book = bookService.searchBook(bookId);

System.***out***.println("\n");

System.***out***.println("<------Search Result------->");

System.***out***.println(book);

}

**private** **void** addSubject() {

Subject subject = **new** Subject();

subject.setSubjectId(Long.*parseLong*(gatValFromConsole("enter subject id")));

subject.setSubTitle(gatValFromConsole("enter subject title"));

subject.setDurationInHours(Integer.*parseInt*(gatValFromConsole("enter duration (In Hours)")));

Subject subjectObj = subjectService.addSubject(subject);

**if** (**null** != subjectObj) {

System.***out***.println("below mentioned subject has been added successfully");

System.***out***.println(subject.toString());

} **else** {

System.***out***.println("unable to add subject");

}

}

**private** **void** deleteSubject() {

**long** subjectId = Long.*parseLong*(gatValFromConsole("enter the subject id to be deleted"));

**if** (subjectService.deleteSubject(subjectId))

System.***out***.println("subject with id :" + subjectId + " has been removed");

}

**private** **void** searchSubject() {

**long** subjectId = Long.*parseLong*(gatValFromConsole("enter the subject id for search"));

Subject subject = subjectService.searchSubject(subjectId);

System.***out***.println("\n");

System.***out***.println("<------Search Result------->");

System.***out***.println(subject);

}

**private** **void** showAll() {

System.***out***.println("\n");

System.***out***.println("<------Total Subject List------->");

subjectService.fetchAllSubject().stream().forEach(System.***out***::println);

System.***out***.println("\n");

System.***out***.println("<------Total Book List------->");

bookService.fetchAllBook().stream().forEach(System.***out***::println);

}

**private** **void** exit() {

System.*exit*(0);

}

**private** String gatValFromConsole(String log) {

System.***out***.println(log);

String inputString = "";

**try** {

BufferedReader bufferRead = **new** BufferedReader(**new** InputStreamReader(System.***in***));

inputString = bufferRead.readLine();

} **catch** (IOException ex) {

ex.printStackTrace();

}

**return** inputString;

}

**public** **void** setSubjectService(SubjectService subjectService) {

**this**.subjectService = subjectService;

}

**public** **void** setBookService(BookService bookService) {

**this**.bookService = bookService;

}

}

**Book.java**

package com.cognizant.model;

import java.io.Serializable;

import java.time.LocalDate;

/\*\*

\* @author Saugata Ray

\*

\*/

public class Book implements Serializable{

private static final long serialVersionUID = 3667779253735136971L;

private long bookId;

private String title;

private double price;

private Integer volume;

private LocalDate publishDate;

public long getBookId() {

return bookId;

}

public void setBookId(long bookId) {

this.bookId = bookId;

}

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public double getPrice() {

return price;

}

public void setPrice(double price) {

this.price = price;

}

public Integer getVolume() {

return volume;

}

public void setVolume(Integer volume) {

this.volume = volume;

}

public LocalDate getPublishDate() {

return publishDate;

}

public void setPublishDate(LocalDate publishDate) {

this.publishDate = publishDate;

}

@Override

public String toString() {

return "Book [bookId=" + bookId + ", title=" + title + ", price=" + price + ", volume=" + volume

+ ", publishDate=" + publishDate + "]";

}

}

**Subject.java**

package com.cognizant.model;

import java.io.Serializable;

import java.util.Set;

/\*\*

\* @author Saugata Ray

\*

\*/

public class Subject implements Serializable{

private static final long serialVersionId = 1L;

private long subjectId;

private String subTitle;

private int durationInHours;

private Set<Book> references;

public long getSubjectId() {

return subjectId;

}

public void setSubjectId(long subjectId) {

this.subjectId = subjectId;

}

public String getSubTitle() {

return subTitle;

}

public void setSubTitle(String subTitle) {

this.subTitle = subTitle;

}

public int getDurationInHours() {

return durationInHours;

}

public void setDurationInHours(int durationInHours) {

this.durationInHours = durationInHours;

}

public Set<Book> getReferences() {

return references;

}

public void setReferences(Set<Book> references) {

this.references = references;

}

@Override

public String toString() {

return "Subject [subjectId=" + subjectId + ", subTitle=" + subTitle + ", durationInHours=" + durationInHours

+ "]";

}

}

**BookService.java**

**package** com.cognizant.service;

**import** java.util.List;

**import** com.cognizant.model.Book;

**public** **interface** BookService {

**public** Book addBook(Book book);

**public** **boolean** deleteBook(**long** bookId);

**public** Book searchBook(**long** bookId);

**public** List<Book> fetchAllBook();

}

**BookServiceImpl.java**

**package** com.cognizant.service;

**import** java.util.List;

**import** com.cognizant.model.Book;

**import** com.cognizant.repository.BookRepository;

**public** **class** BookServiceImpl **implements** BookService {

**private** BookRepository bookRepository;

**public** **void** setBookRepository(BookRepository bookRepository) {

**this**.bookRepository = bookRepository;

}

@Override

**public** Book addBook(Book book) {

**return** bookRepository.addBook(book);

}

@Override

**public** **boolean** deleteBook(**long** bookId) {

**return** bookRepository.deleteBook(bookId);

}

@Override

**public** Book searchBook(**long** bookId) {

**return** bookRepository.searchBook(bookId);

}

@Override

**public** List<Book> fetchAllBook() {

**return** bookRepository.fetchAllBook();

}

}

**SubjectService.java**

**package** com.cognizant.service;

**import** java.util.List;

**import** com.cognizant.model.Subject;

**public** **interface** SubjectService {

**public** Subject addSubject(Subject subject);

**public** **boolean** deleteSubject(**long** subjectId);

**public** Subject searchSubject(**long** subjectId);

**public** List<Subject> fetchAllSubject();

}

**SubjectServiceImpl.java**

**package** com.cognizant.service;

**import** java.util.List;

**import** com.cognizant.model.Subject;

**import** com.cognizant.repository.SubjectRepository;

**public** **class** SubjectServiceImpl **implements** SubjectService{

**private** SubjectRepository subjectRepository;

**public** **void** setSubjectRepository(SubjectRepository subjectRepository) {

**this**.subjectRepository = subjectRepository;

}

@Override

**public** Subject addSubject(Subject subject) {

**return** subjectRepository.addSubject(subject);

}

@Override

**public** **boolean** deleteSubject(**long** subjectId) {

**return** subjectRepository.deleteSubject(subjectId);

}

@Override

**public** Subject searchSubject(**long** subjectId) {

**return** subjectRepository.searchSubject(subjectId);

}

@Override

**public** List<Subject> fetchAllSubject() {

**return** subjectRepository.fetchAllSubject();

}

}

**BookRepository.java**

**package** com.cognizant.repository;

**import** java.util.List;

**import** com.cognizant.model.Book;

**public** **interface** BookRepository {

**public** Book addBook(Book book);

**public** **boolean** deleteBook(**long** bookId);

**public** Book searchBook(**long** bookId);

**public** List<Book> fetchAllBook();

}

**BookRepositoryImpl.java**

**package** com.cognizant.repository;

**import** java.time.LocalDate;

**import** java.util.ArrayList;

**import** java.util.List;

**import** com.cognizant.model.Book;

**public** **class** BookRepositoryImpl **implements** BookRepository{

**private** List<Book> bookList=**new** ArrayList<Book>();

@Override

**public** Book addBook(Book book) {

**return** bookList.add(book)?book:**null**;

}

@Override

**public** **boolean** deleteBook(**long** bookId) {

**return** bookList.removeIf(book->book.getBookId()==bookId);

}

@Override

**public** Book searchBook(**long** bookId) {

Book book= bookList.stream()

.filter(b -> b.getBookId() == bookId).reduce((a, b) -> {

**throw** **new** IllegalStateException("Multiple elements: " + a + ", " + b);

}).get();

**return** book;

}

@Override

**public** List<Book> fetchAllBook() {

**return** bookList;

}

}

**SubjectRepository.java**

**package** com.cognizant.repository;

**import** java.util.List;

**import** com.cognizant.model.Subject;

**public** **interface** SubjectRepository {

**public** Subject addSubject(Subject subject);

**public** **boolean** deleteSubject(**long** subjectId);

**public** Subject searchSubject(**long** subjectId);

**public** List<Subject> fetchAllSubject();

}

**SubjectRepositoryImpl.java**

**package** com.cognizant.repository;

**import** java.util.ArrayList;

**import** java.util.List;

**import** com.cognizant.model.Subject;

**public** **class** SubjectRepositoryImpl **implements** SubjectRepository{

**private** List<Subject> subjectList=**new** ArrayList<Subject>();

@Override

**public** Subject addSubject(Subject subject) {

**return** subjectList.add(subject)?subject:**null**;

}

@Override

**public** **boolean** deleteSubject(**long** subjectId) {

**return** subjectList.removeIf(sub->sub.getSubjectId()==subjectId);

}

@Override

**public** Subject searchSubject(**long** subjectId) {

Subject subject= subjectList.stream()

.filter(sub -> sub.getSubjectId() == subjectId).reduce((a, b) -> {

**throw** **new** IllegalStateException("Multiple elements: " + a + ", " + b);

}).get();

**return** subject;

}

@Override

**public** List<Subject> fetchAllSubject() {

**return** subjectList;

}

}

**applicationContext.xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:context=*"http://www.springframework.org/schema/context"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context.xsd"*>

<bean id=*"subjectService"* class=*"com.cognizant.service.SubjectServiceImpl"*>

<property name=*"subjectRepository"* ref=*"subjectRepository"* />

</bean>

<bean id=*"subjectRepository"* class=*"com.cognizant.repository.SubjectRepositoryImpl"* />

<bean id=*"bookService"* class=*"com.cognizant.service.BookServiceImpl"*>

<property name=*"bookRepository"* ref=*"bookRepository"* />

</bean>

<bean id=*"bookRepository"* class=*"com.cognizant.repository.BookRepositoryImpl"* />

<bean id=*"commandLineRunner"* class=*"com.cognizant.CommandLineRunner"*>

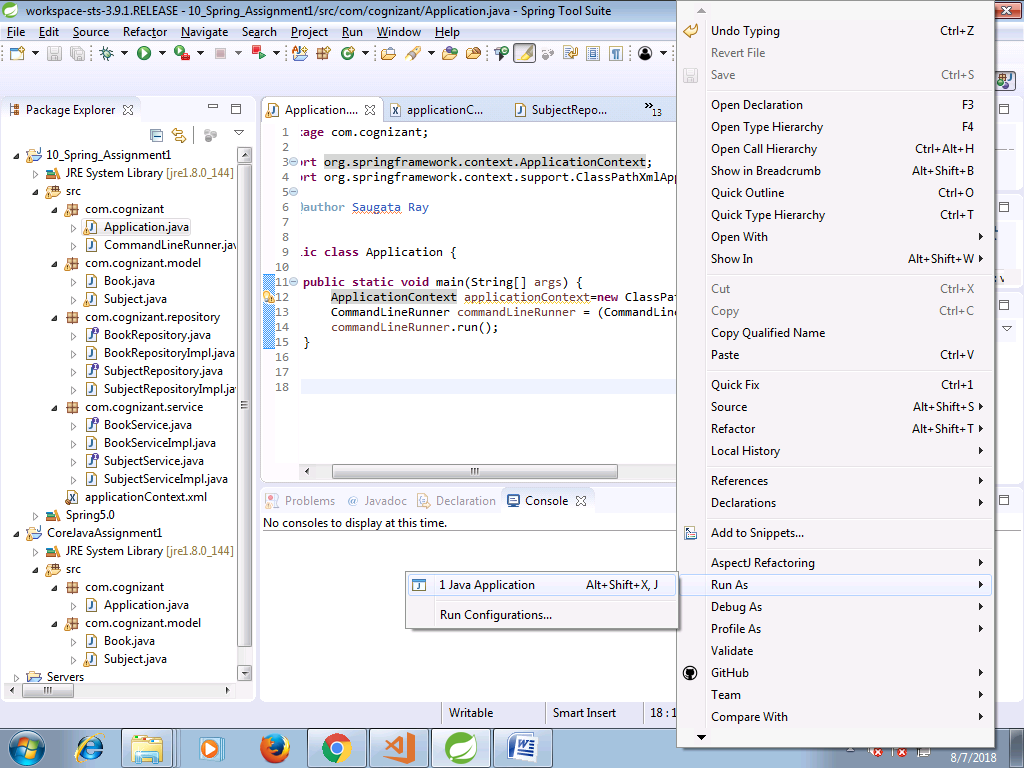
<property name=*"subjectService"* ref=*"subjectService"* />

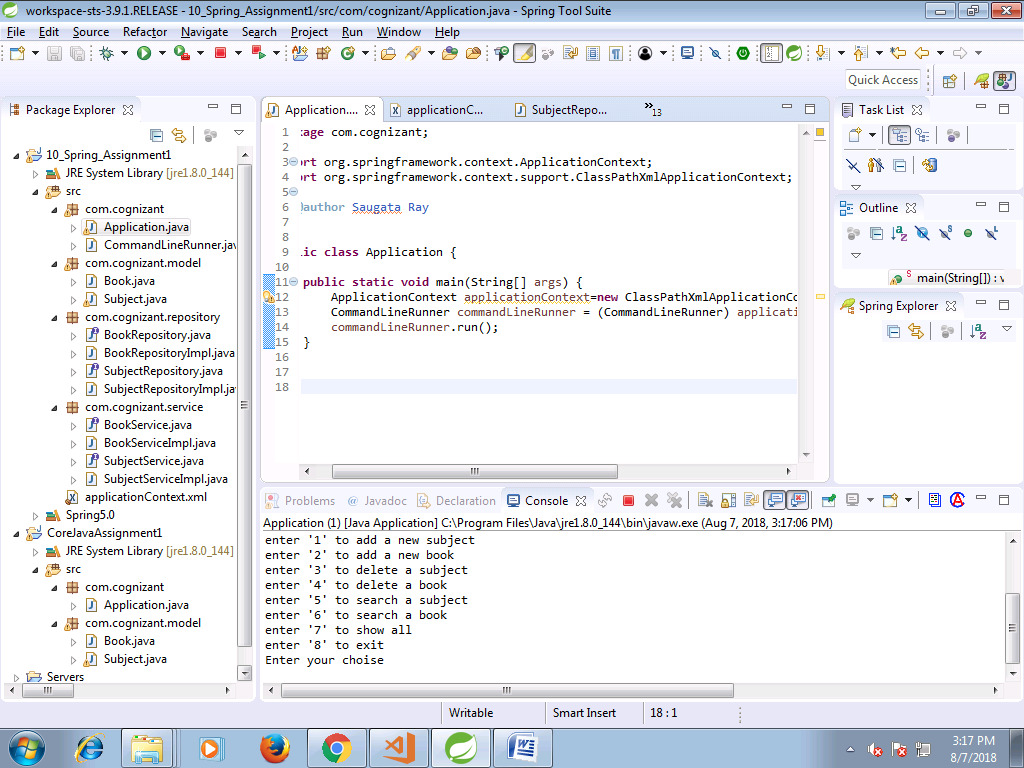
<property name=*"bookService"* ref=*"bookService"* />

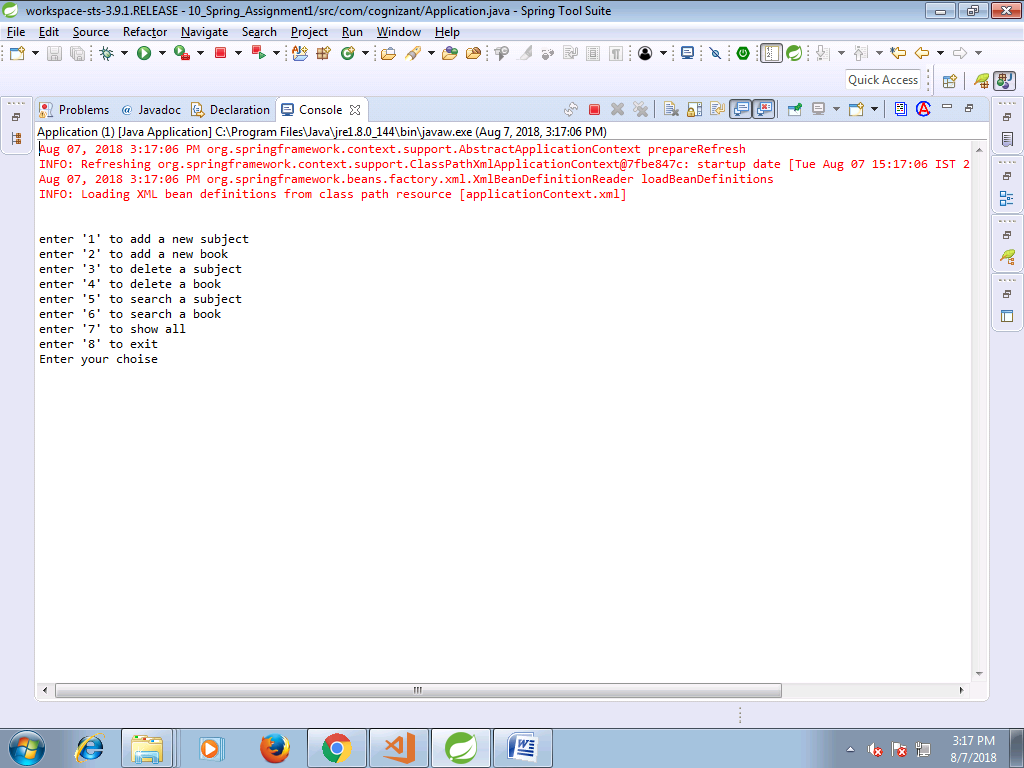
</bean>

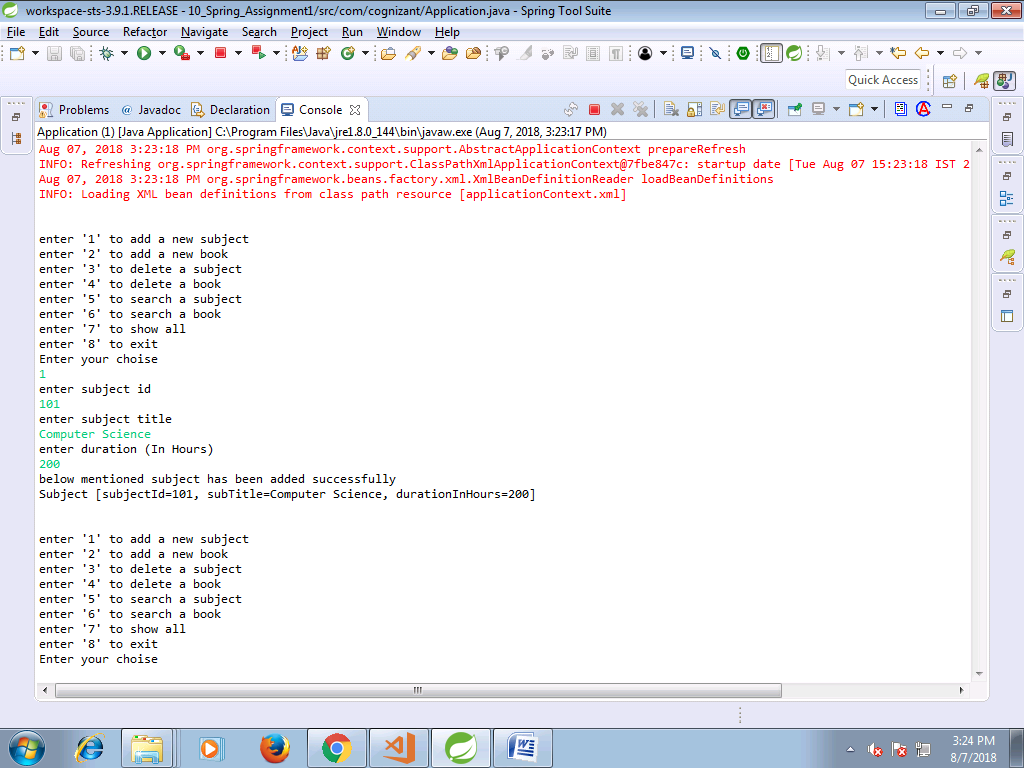
</beans>

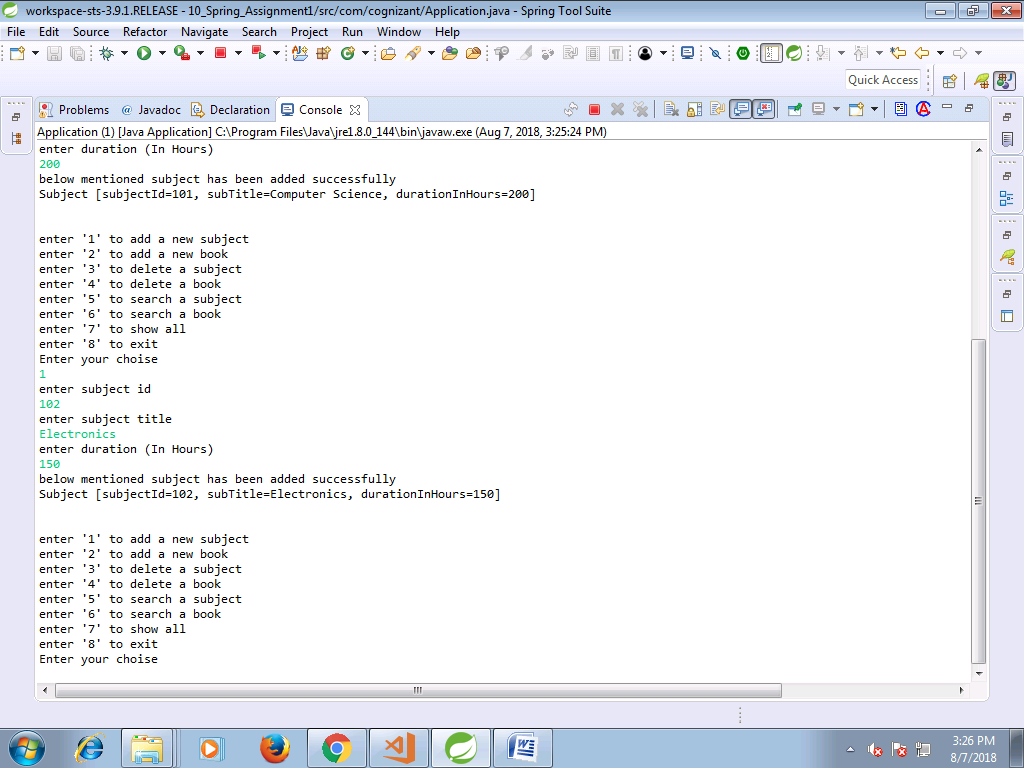
**output:**

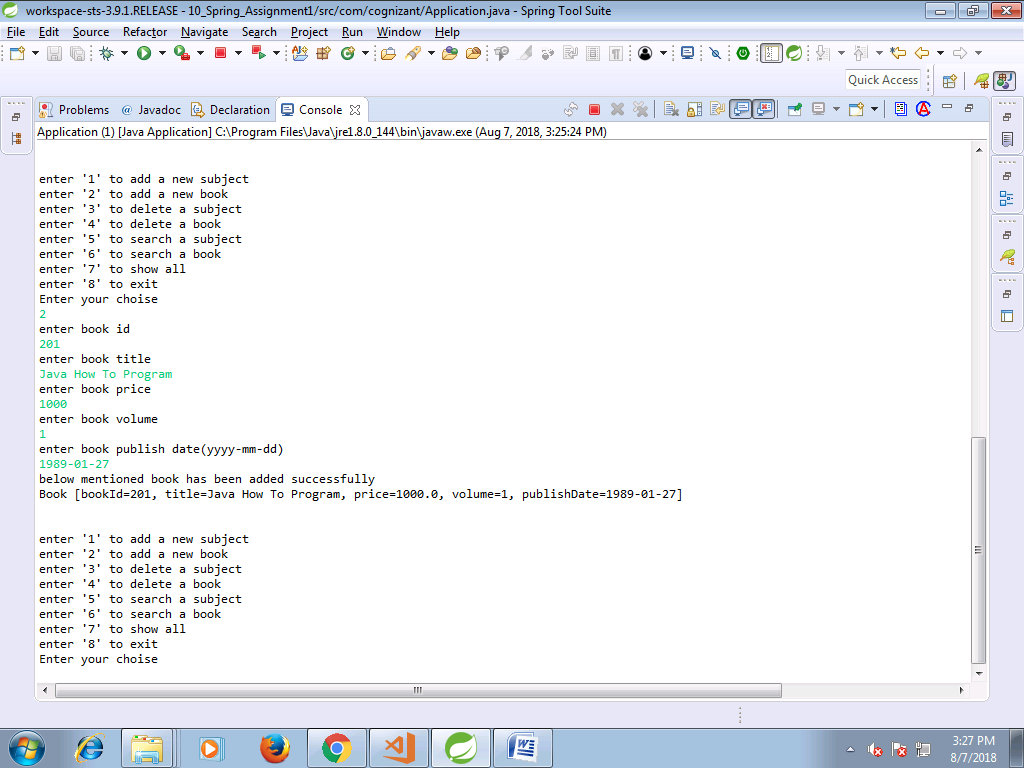
****

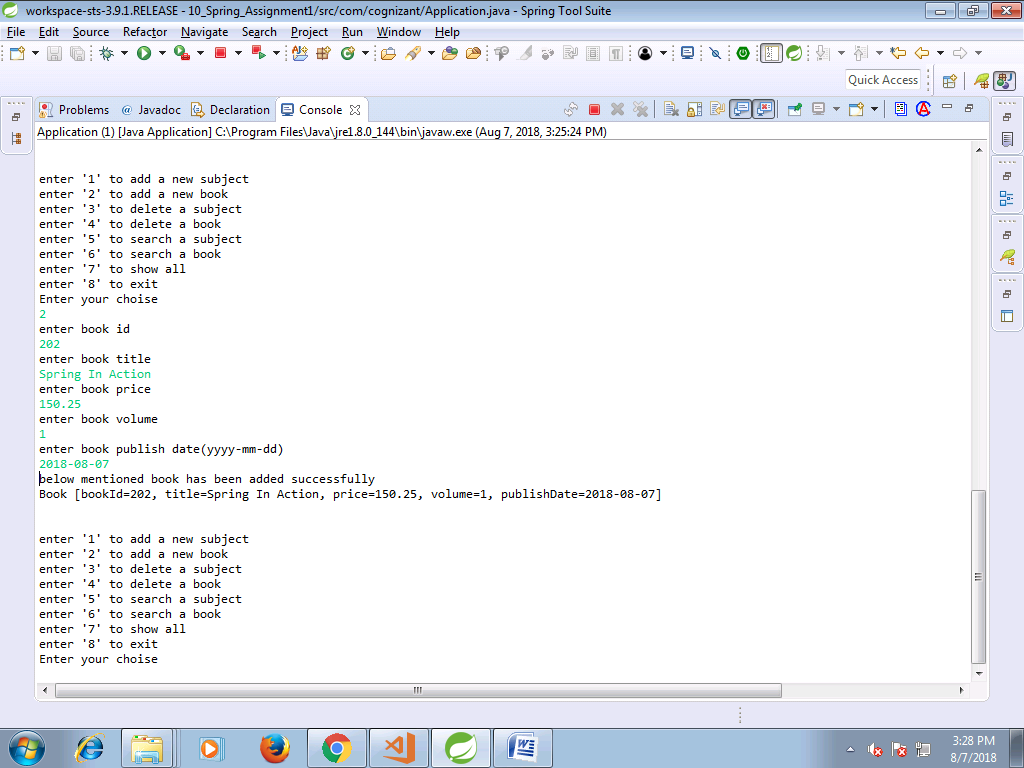
****

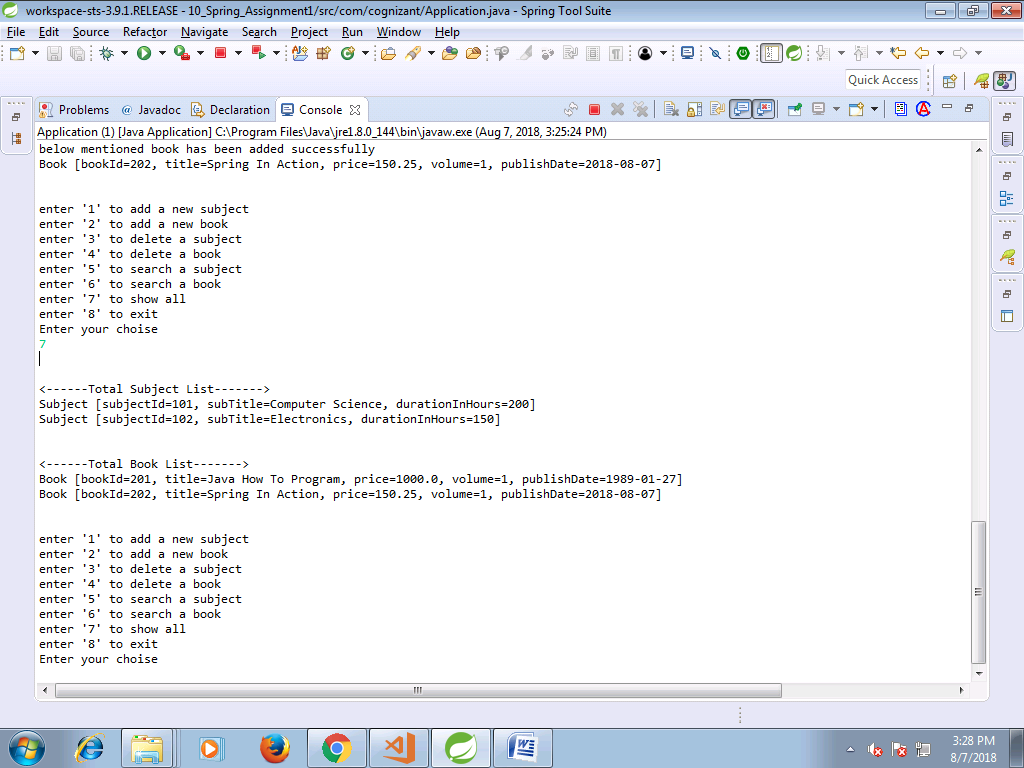
****

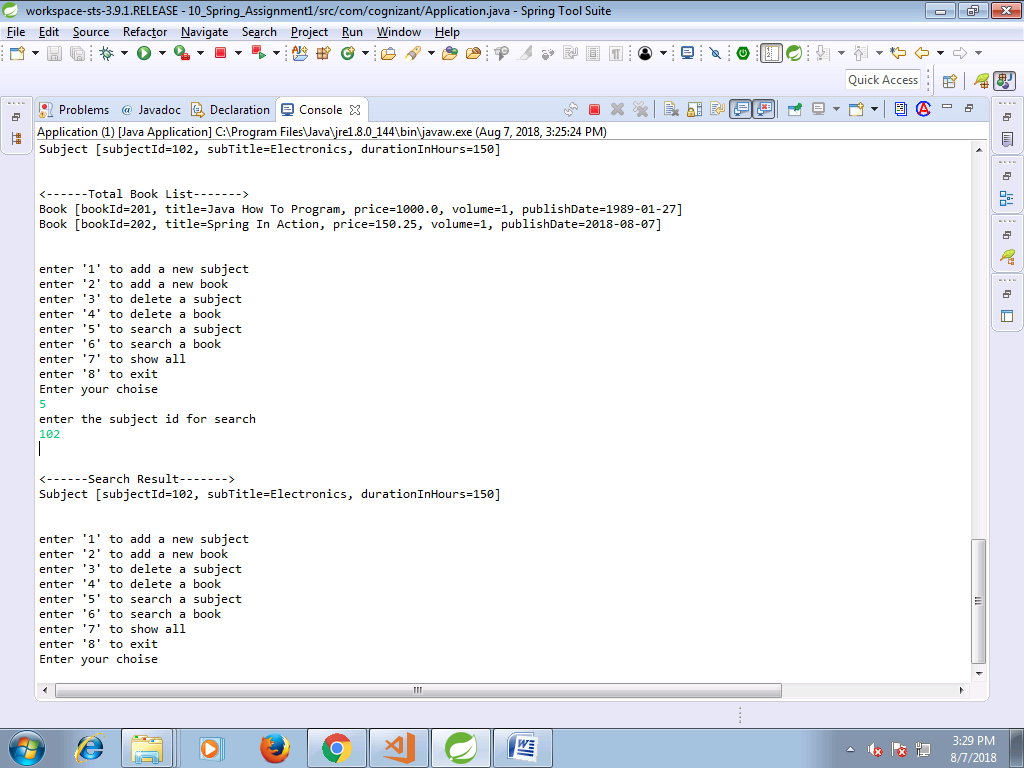
****

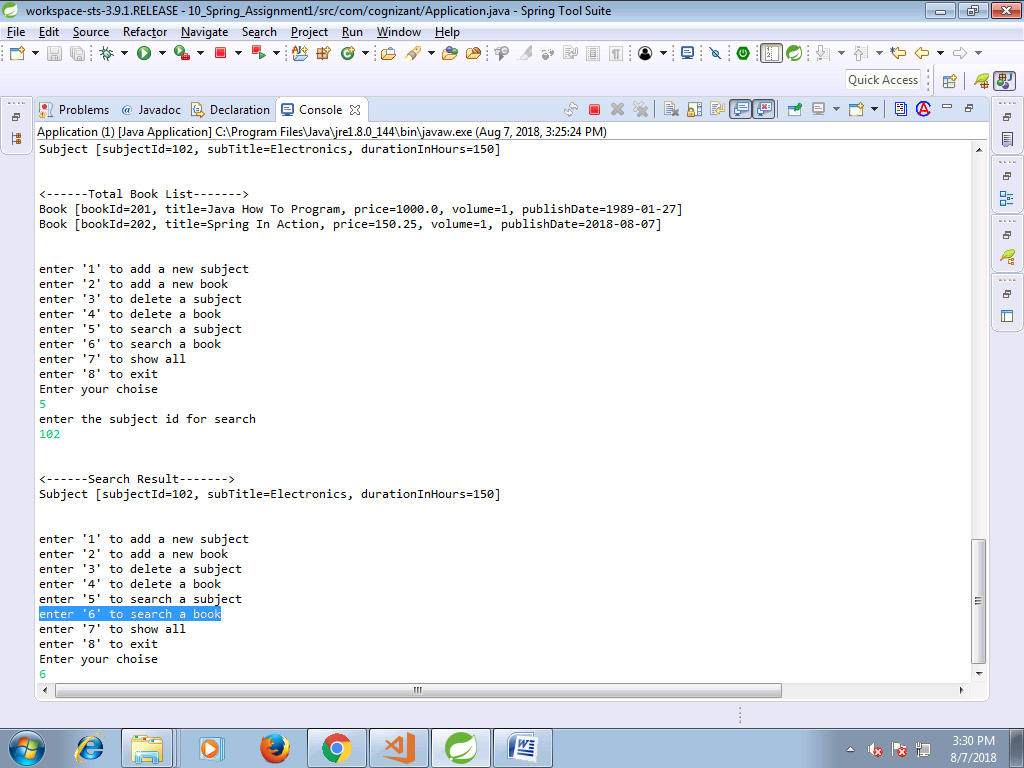
****

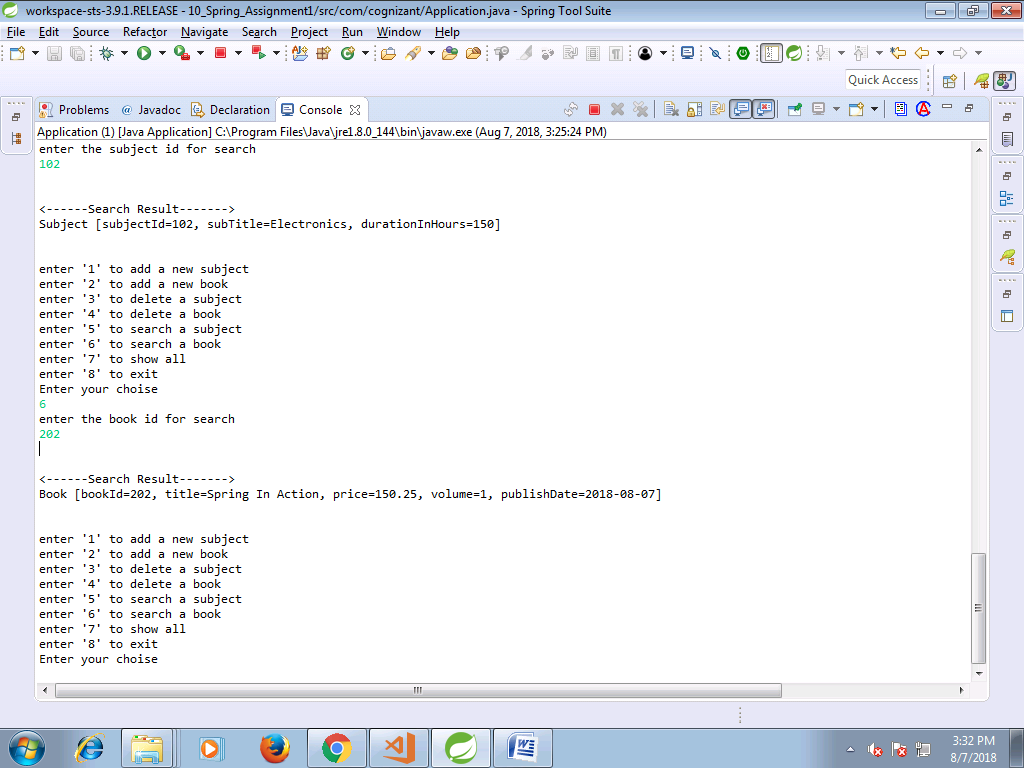
****

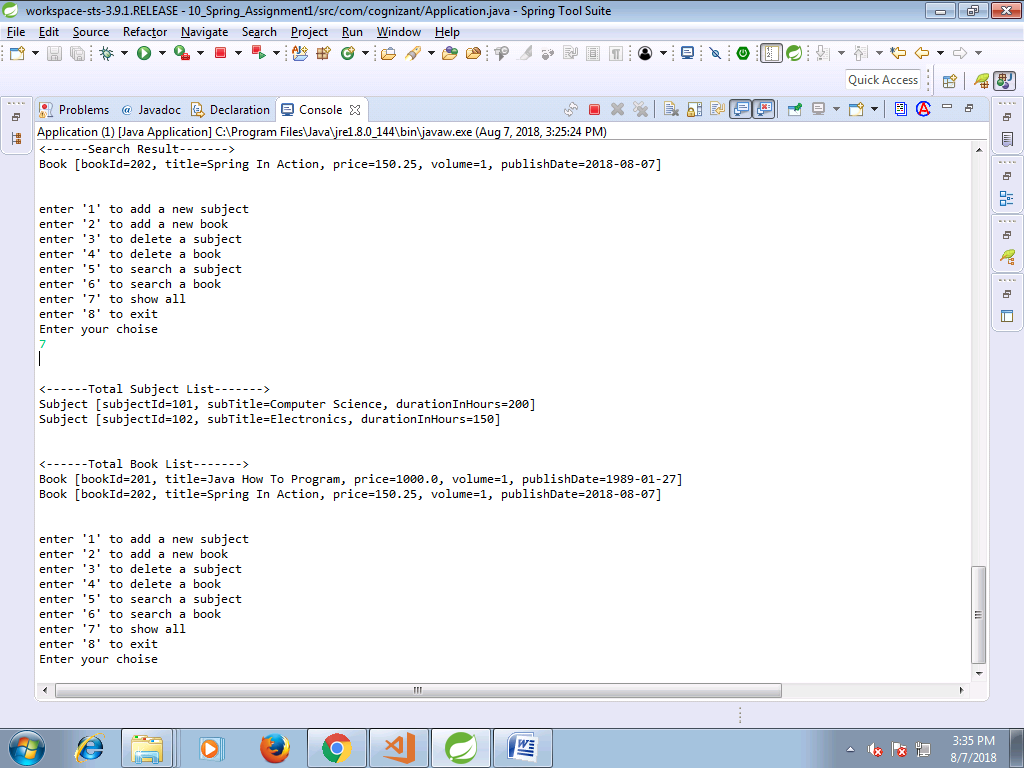
****

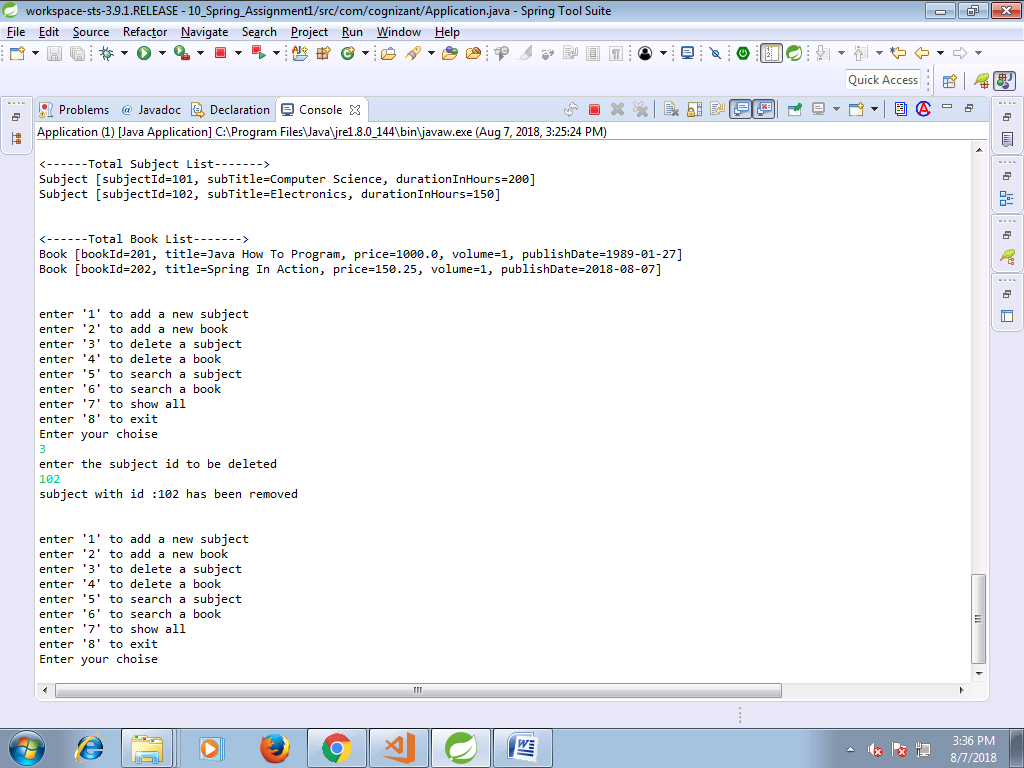
****

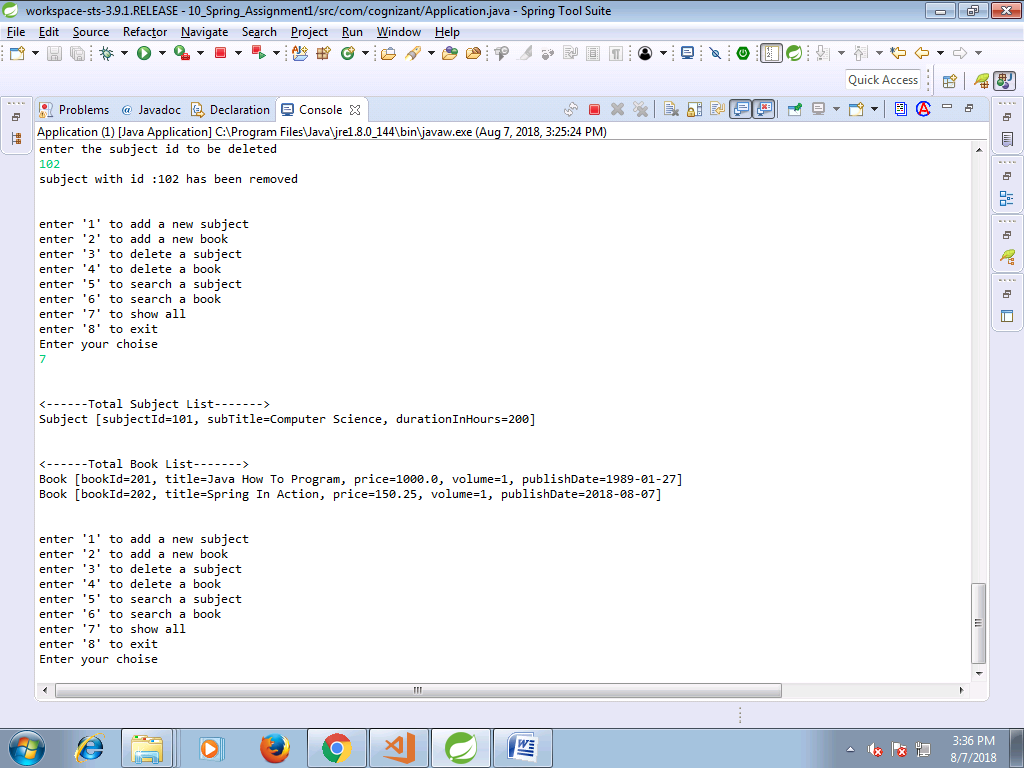
****

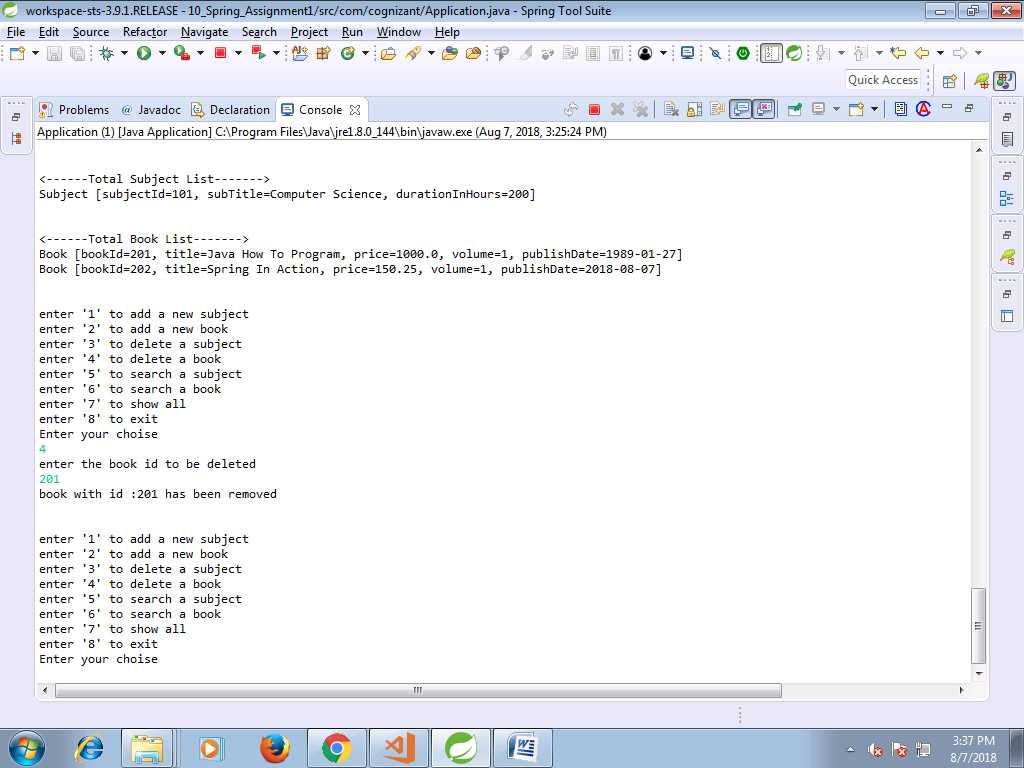
****

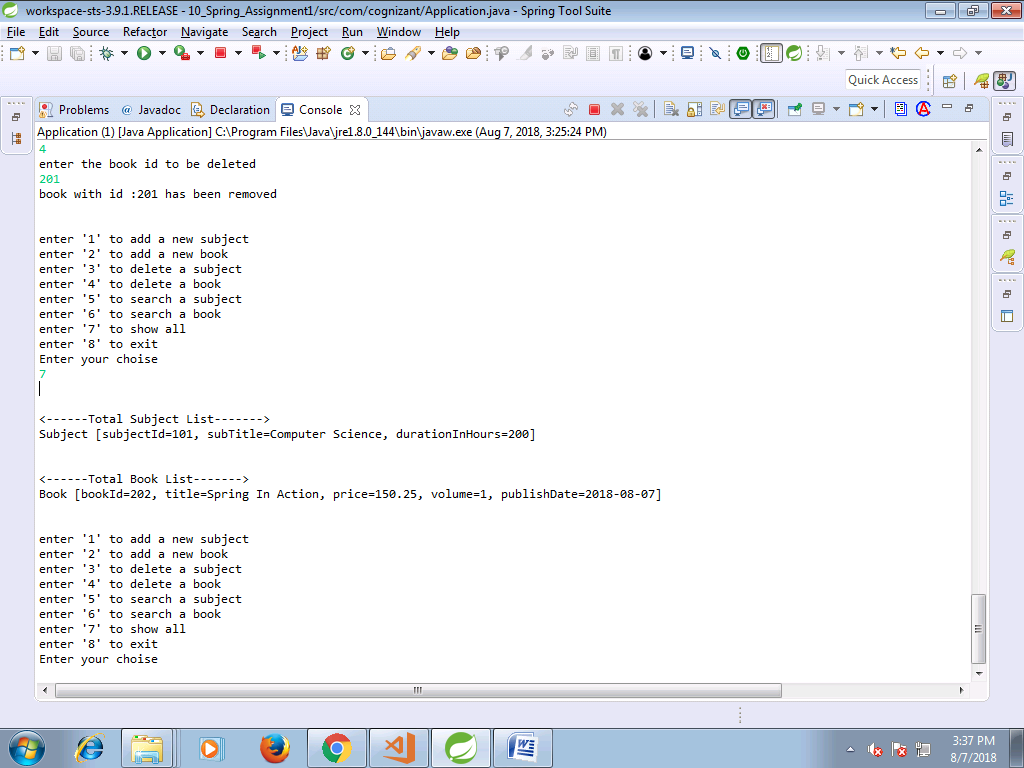
****

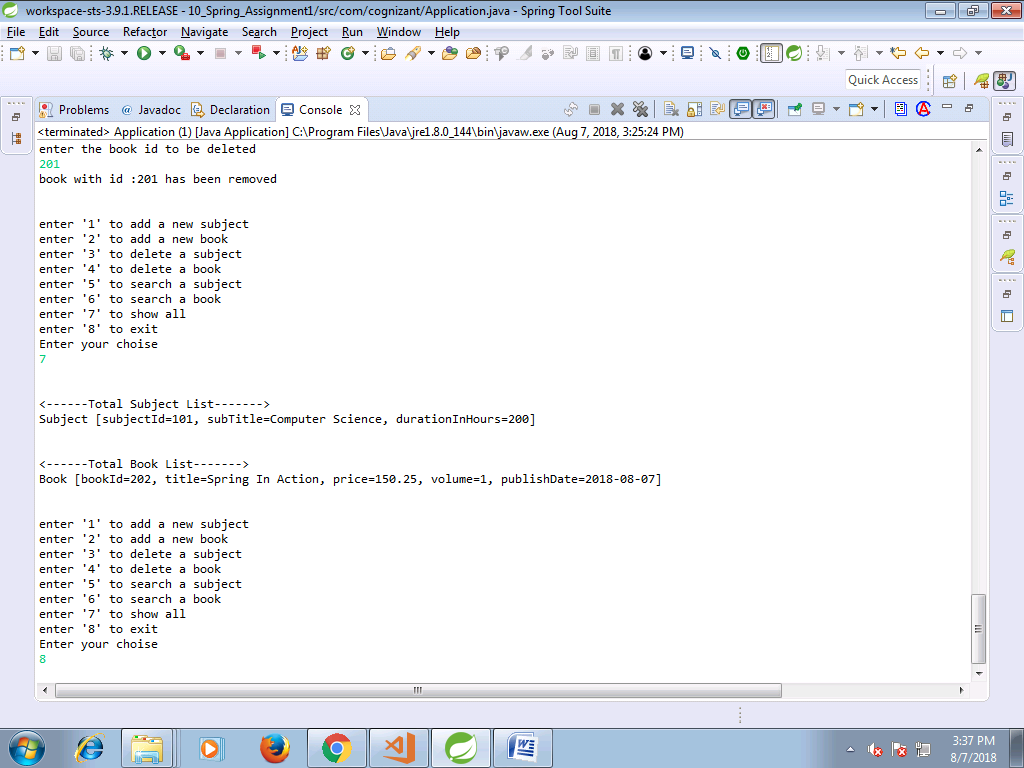
****

****

****

****

****

****