**14\_SpringTest\_Assignment1**

com.cognizant.fsd.springtestassignment1.Application

**package** com.cognizant.fsd.springtestassignment1;

**import** **static** org.mockito.Mockito.*when*;

**import** java.util.Optional;

**import** org.junit.Assert;

**import** org.junit.Test;

**import** org.junit.runner.RunWith;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.boot.test.context.SpringBootTest;

**import** org.springframework.boot.test.mock.mockito.MockBean;

**import** org.springframework.test.context.junit4.SpringRunner;

**import** com.cognizant.fsd.springtestassignment1.model.Subject;

**import** com.cognizant.fsd.springtestassignment1.repository.SubjectRepository;

**import** com.cognizant.fsd.springtestassignment1.service.SubjectService;

@RunWith(SpringRunner.**class**)

@SpringBootTest

**public** **class** ApplicationTests {

@MockBean

**private** SubjectRepository subjectRepository;

@Autowired

**private** SubjectService subjectService;

@Test

**public** **void** subjectServiceSearchSubjectTest() {

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

subject.setSubjectId(1l);

Optional<Subject> optionalsubject = Optional.*ofNullable*(subject);

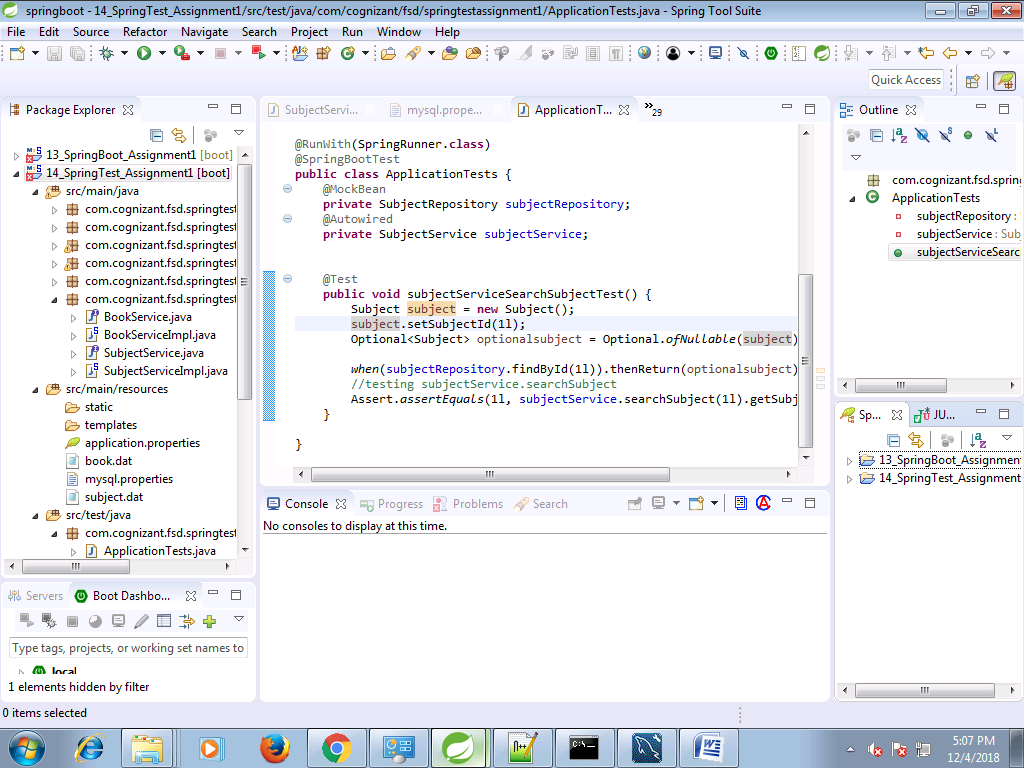
*when*(subjectRepository.findById(1l)).thenReturn(optionalsubject);

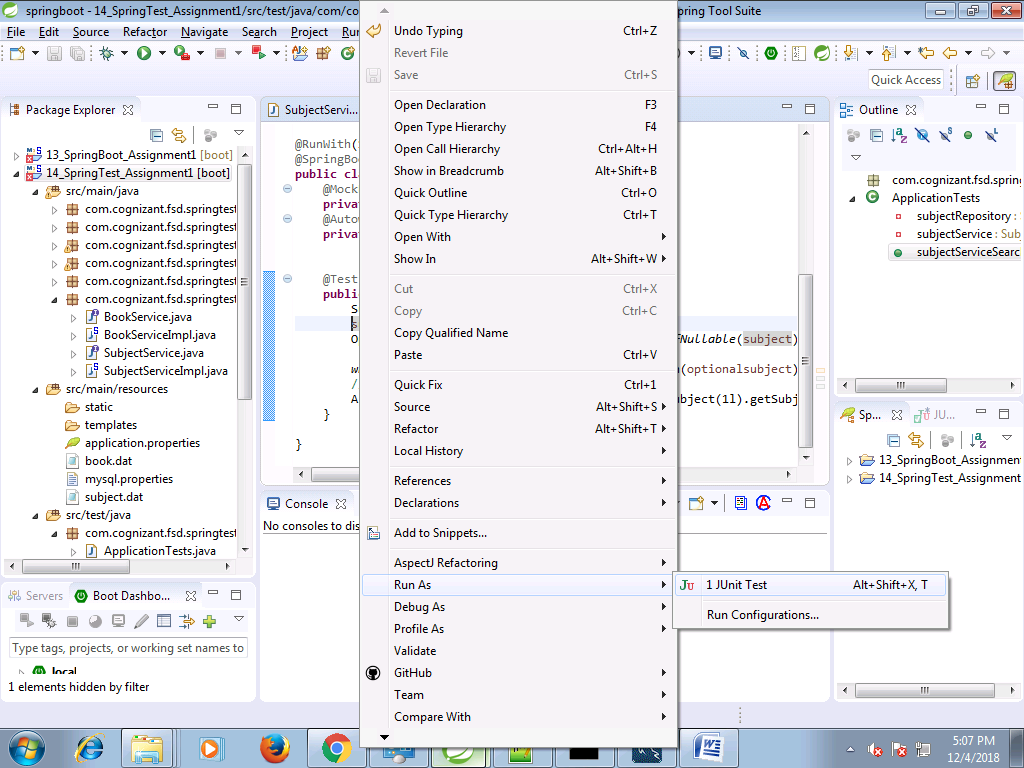
//testing subjectService.searchSubject

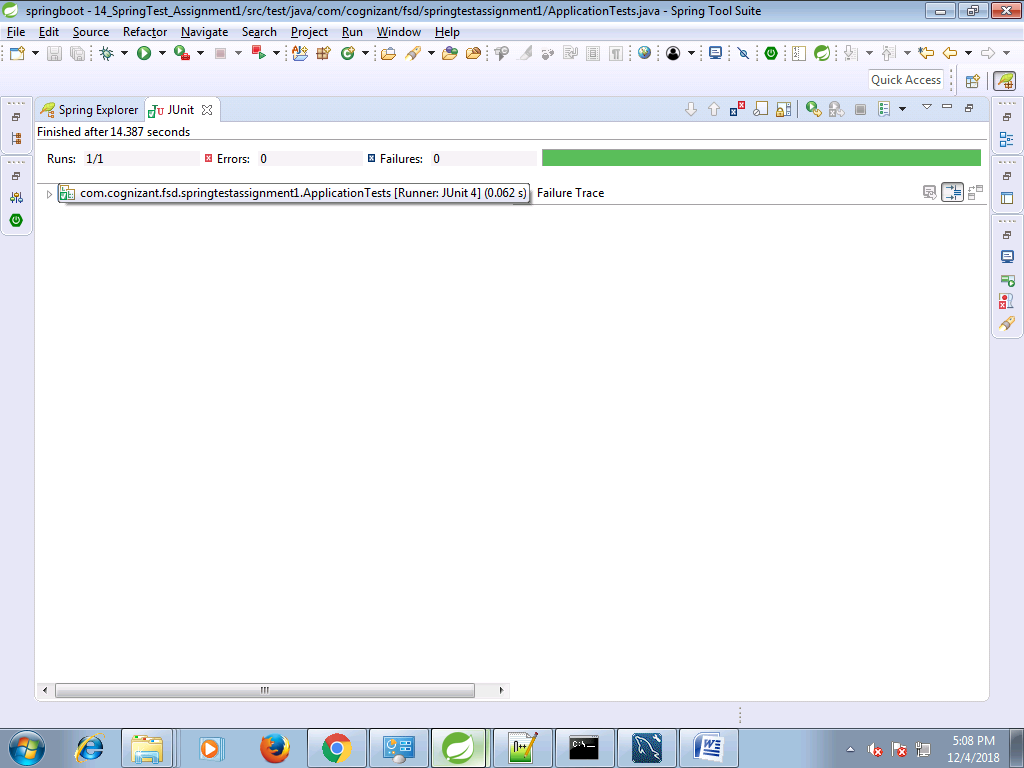
Assert.*assertEquals*(1l, subjectService.searchSubject(1l).getSubjectId().longValue());

}

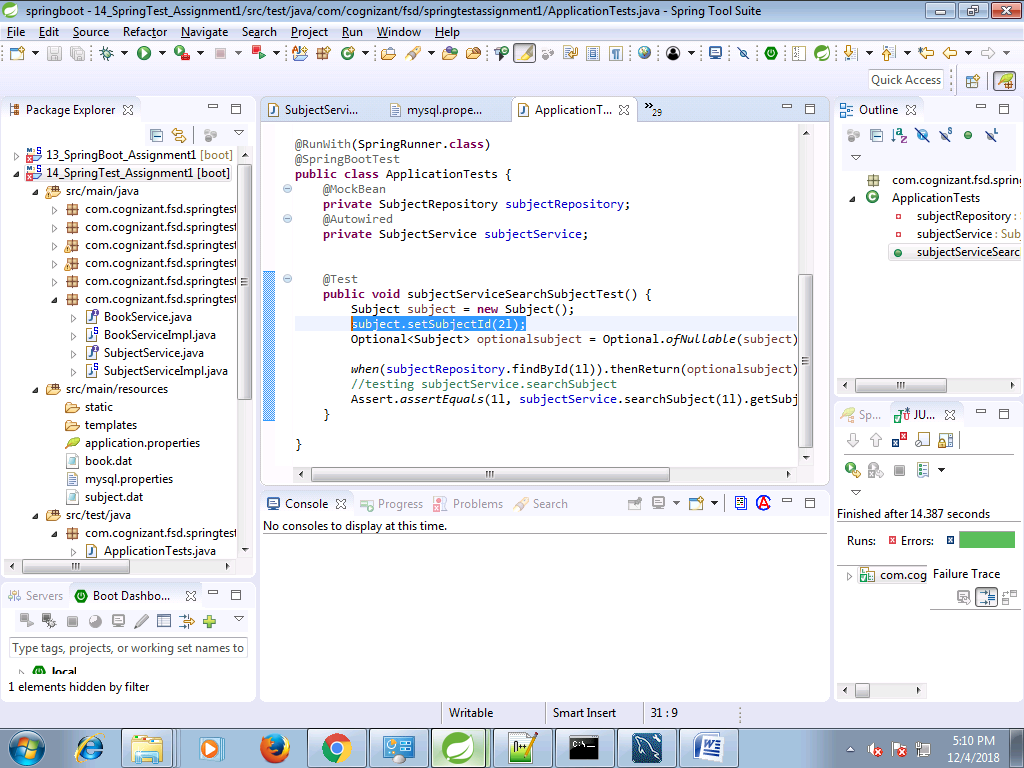
}

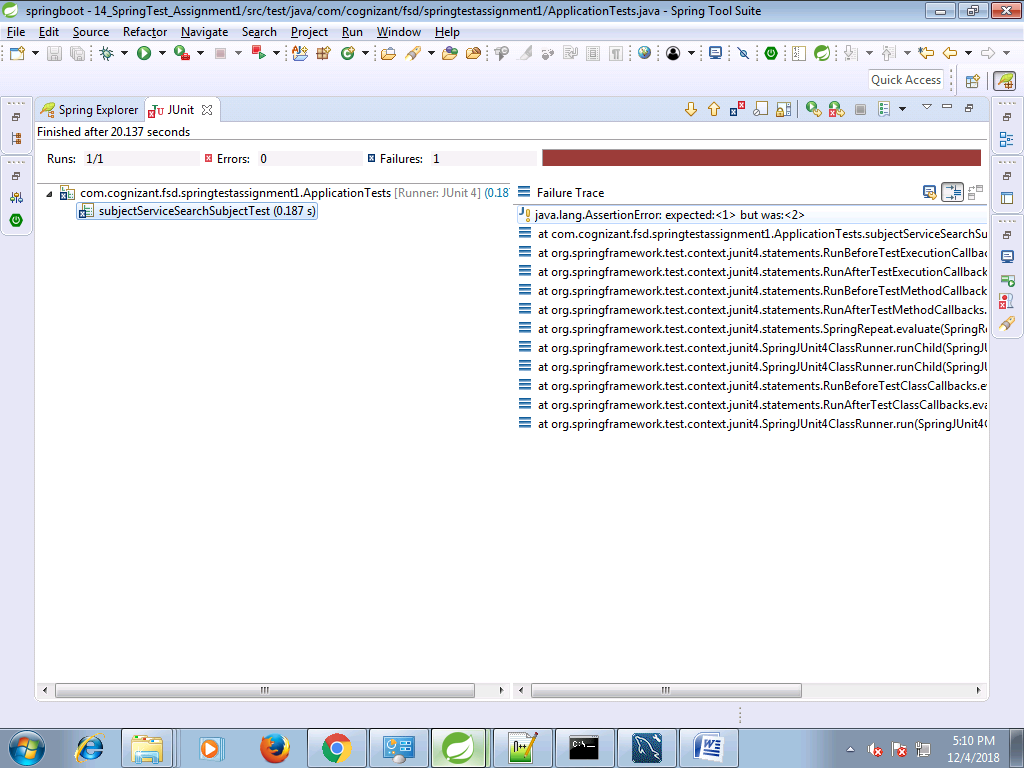






**negative testing :**

****

****

**Testing subjectService.addSubject**

**package** com.cognizant.fsd.springtestassignment1;

**import** **static** org.mockito.Mockito.*when*;

**import** java.util.Optional;

**import** org.junit.Assert;

**import** org.junit.Test;

**import** org.junit.runner.RunWith;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.boot.test.context.SpringBootTest;

**import** org.springframework.boot.test.mock.mockito.MockBean;

**import** org.springframework.test.context.junit4.SpringRunner;

**import** com.cognizant.fsd.springtestassignment1.model.Subject;

**import** com.cognizant.fsd.springtestassignment1.repository.SubjectRepository;

**import** com.cognizant.fsd.springtestassignment1.service.SubjectService;

@RunWith(SpringRunner.**class**)

@SpringBootTest

**public** **class** ApplicationTests {

@MockBean

**private** SubjectRepository subjectRepository;

@Autowired

**private** SubjectService subjectService;

@Test

**public** **void** subjectServiceAddSubjectTest() {

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

subject.setSubjectId(1l);

subject.setSubTitle("Computer");

*when*(subjectRepository.save(subject)).thenReturn(subject);

//testing subjectService.addSubject

Assert.*assertEquals*(subject, subjectService.addSubject(subject));

}

//@Test

**public** **void** subjectServiceSearchSubjectTest() {

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

subject.setSubjectId(2l);

Optional<Subject> optionalsubject = Optional.*ofNullable*(subject);

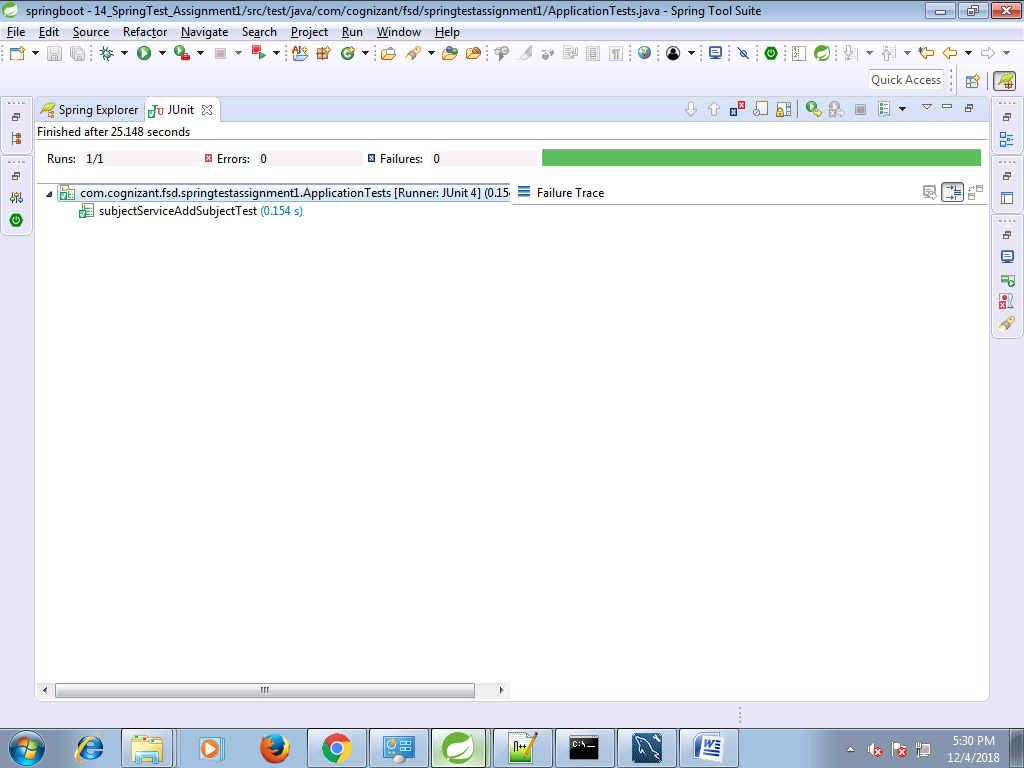
*when*(subjectRepository.findById(1l)).thenReturn(optionalsubject);

//testing subjectService.searchSubject

Assert.*assertEquals*(1l, subjectService.searchSubject(1l).getSubjectId().longValue());

}

}

****

**negative testing**

@Test

**public** **void** subjectServiceAddSubjectTest() {

Subject subjectObject = **new** Subject();

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

subject.setSubjectId(1l);

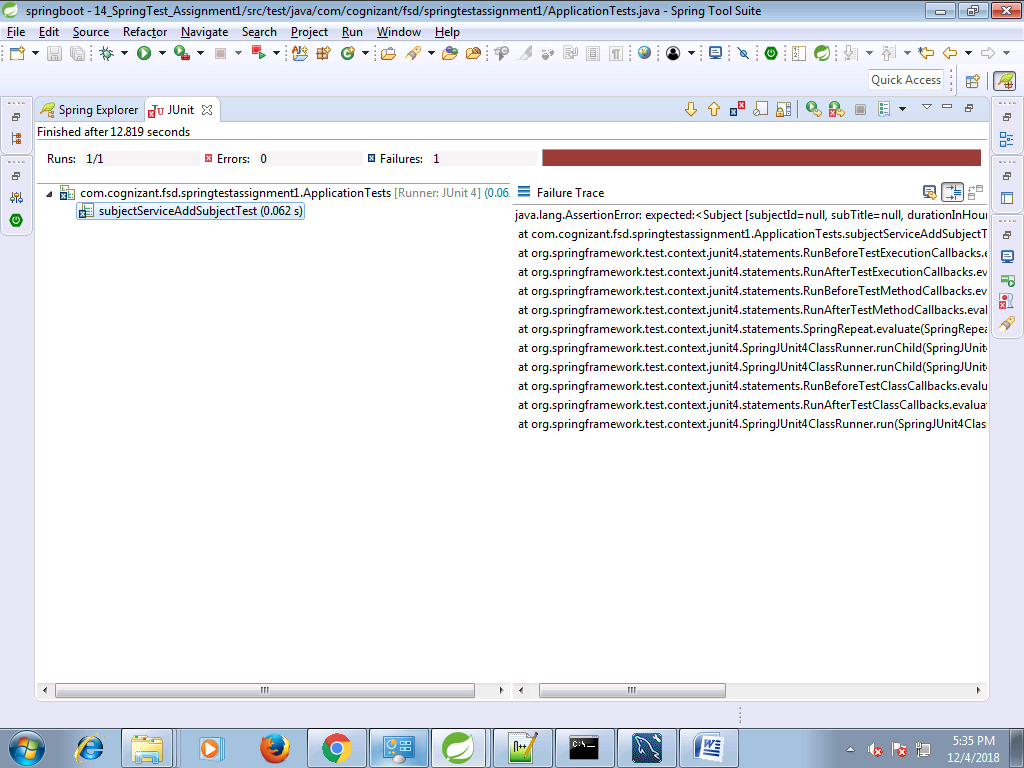
subject.setSubTitle("Computer");

*when*(subjectRepository.save(subject)).thenReturn(subject);

//testing subjectService.addSubject

Assert.*assertEquals*(subjectObject, subjectService.addSubject(subject));

}



**Testing subjectService.addSubject**

**package** com.cognizant.fsd.springtestassignment1;

**import** **static** org.mockito.Mockito.*when*;

**import** java.util.ArrayList;

**import** java.util.List;

**import** java.util.Optional;

**import** org.junit.Assert;

**import** org.junit.Test;

**import** org.junit.runner.RunWith;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.boot.test.context.SpringBootTest;

**import** org.springframework.boot.test.mock.mockito.MockBean;

**import** org.springframework.test.context.junit4.SpringRunner;

**import** com.cognizant.fsd.springtestassignment1.model.Subject;

**import** com.cognizant.fsd.springtestassignment1.repository.SubjectRepository;

**import** com.cognizant.fsd.springtestassignment1.service.SubjectService;

@RunWith(SpringRunner.**class**)

@SpringBootTest

**public** **class** ApplicationTests {

@MockBean

**private** SubjectRepository subjectRepository;

@Autowired

**private** SubjectService subjectService;

@Test

**public** **void** subjectServiceFetchAllSubjectTest() {

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

Subject subjectObject = **new** Subject();

subjectObject.setSubjectId(1l);

subjectObject.setSubTitle("Bengali");

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

subject.setSubjectId(1l);

subject.setSubTitle("Computer");

subjectList.add(subjectObject);

subjectList.add(subject);

*when*(subjectRepository.findAll()).thenReturn(subjectList);

//testing subjectService.fetchAllSubject

Assert.*assertEquals*(subjectList, subjectService.fetchAllSubject());

}

//@Test

**public** **void** subjectServiceAddSubjectTest() {

Subject subjectObject = **new** Subject();

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

subject.setSubjectId(1l);

subject.setSubTitle("Computer");

*when*(subjectRepository.save(subject)).thenReturn(subject);

//testing subjectService.addSubject

Assert.*assertEquals*(subjectObject, subjectService.addSubject(subject));

}

//@Test

**public** **void** subjectServiceSearchSubjectTest() {

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

subject.setSubjectId(2l);

Optional<Subject> optionalsubject = Optional.*ofNullable*(subject);

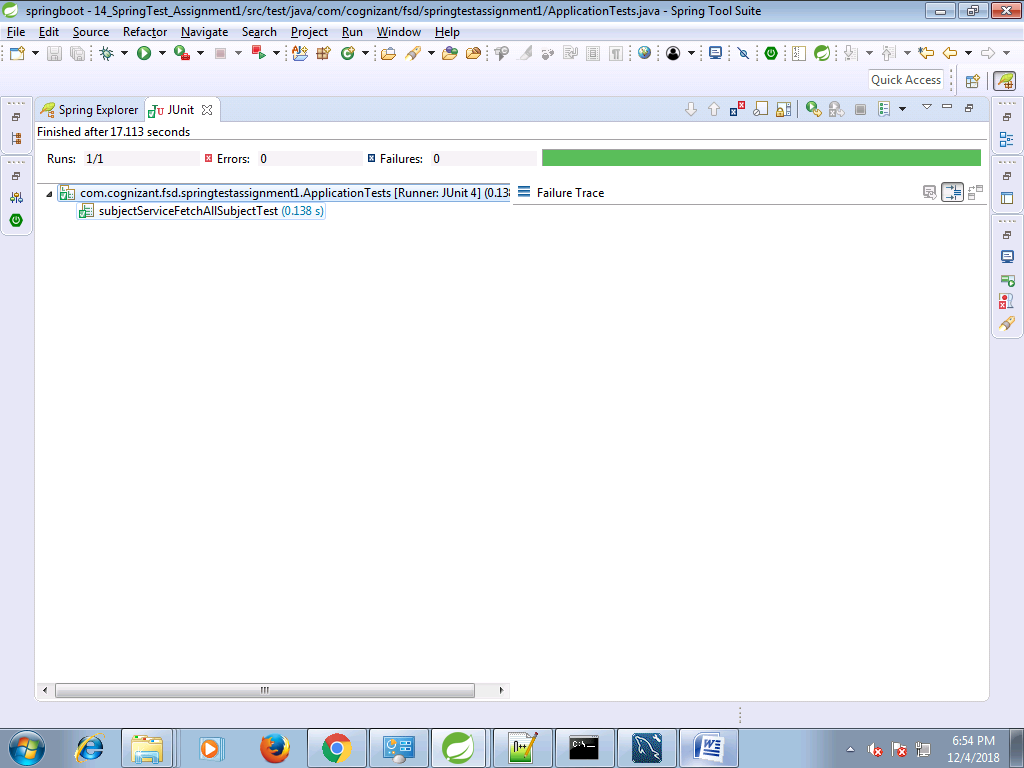
*when*(subjectRepository.findById(1l)).thenReturn(optionalsubject);

//testing subjectService.searchSubject

Assert.*assertEquals*(1l, subjectService.searchSubject(1l).getSubjectId().longValue());

}

}



**negative testing**

@Test

**public** **void** subjectServiceFetchAllSubjectTest() {

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

Subject subjectObject = **new** Subject();

subjectObject.setSubjectId(1l);

subjectObject.setSubTitle("Bengali");

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

subject.setSubjectId(1l);

subject.setSubTitle("Computer");

subjectList.add(subjectObject);

subjectList.add(subject);

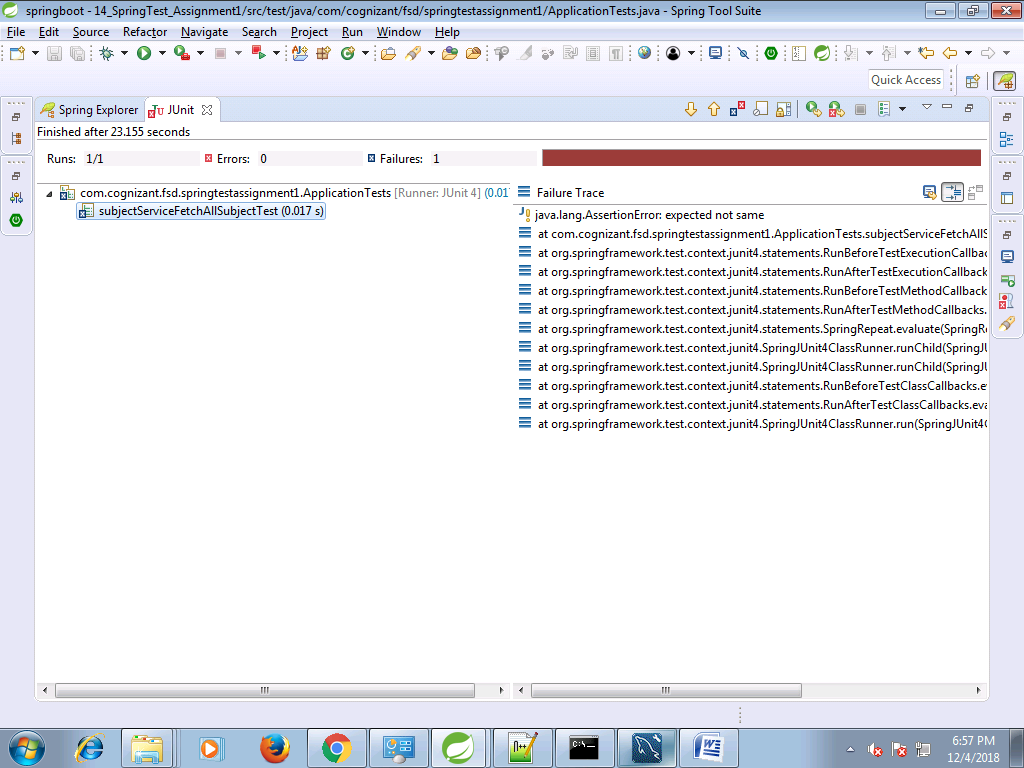
*when*(subjectRepository.findAll()).thenReturn(subjectList);

//testing subjectService.fetchAllSubject

//Assert.assertEquals(subjectList, subjectService.fetchAllSubject());

Assert.*assertNotSame*(subjectList, subjectService.fetchAllSubject());

}

****

**BookService Testing**

**package** com.cognizant.fsd.springtestassignment1.service;

**import** java.util.List;

**import** com.cognizant.fsd.springtestassignment1.model.Book;

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

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

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

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

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

}

**package** com.cognizant.fsd.springtestassignment1.service;

**import** java.util.List;

**import** java.util.Optional;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Service;

**import** org.springframework.transaction.annotation.Transactional;

**import** com.cognizant.fsd.springtestassignment1.model.Book;

**import** com.cognizant.fsd.springtestassignment1.repository.BookRepository;

@Service("bookService")

@Transactional

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

@Autowired

**private** BookRepository bookRepository;

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

**this**.bookRepository = bookRepository;

}

@Override

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

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

}

@Override

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

bookRepository.deleteById(bookId);

**return** **true**;

}

@Override

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

Optional<Book> book=bookRepository.findById(bookId);

**return** book!=**null**?book.get():**null**;

}

@Override

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

**return** (List<Book>) bookRepository.findAll();

}

}

**package** com.cognizant.fsd.springtestassignment1;

**import** **static** org.mockito.Mockito.*when*;

**import** java.util.ArrayList;

**import** java.util.List;

**import** java.util.Optional;

**import** org.junit.Assert;

**import** org.junit.Test;

**import** org.junit.runner.RunWith;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.boot.test.context.SpringBootTest;

**import** org.springframework.boot.test.mock.mockito.MockBean;

**import** org.springframework.test.context.junit4.SpringRunner;

**import** com.cognizant.fsd.springtestassignment1.model.Book;

**import** com.cognizant.fsd.springtestassignment1.model.Subject;

**import** com.cognizant.fsd.springtestassignment1.repository.BookRepository;

**import** com.cognizant.fsd.springtestassignment1.repository.SubjectRepository;

**import** com.cognizant.fsd.springtestassignment1.service.BookService;

**import** com.cognizant.fsd.springtestassignment1.service.SubjectService;

@RunWith(SpringRunner.**class**)

@SpringBootTest

**public** **class** ApplicationTests {

@MockBean

**private** SubjectRepository subjectRepository;

@MockBean

**private** BookRepository bookRepository;

@Autowired

**private** SubjectService subjectService;

@Autowired

**private** BookService bookService;

@Test

**public** **void** bookServiceSearchSubjectTest() {

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

book.setBookId(201l);

book.setTitle("Java How To Program");

Optional<Book> optionalBook = Optional.*ofNullable*(book);

*when*(bookRepository.findById(201l)).thenReturn(optionalBook);

//testing bookService.searchBook

Assert.*assertEquals*(book, bookService.searchBook(201l));

}

//@Test

**public** **void** bookServiceAddSubjectTest() {

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

book.setBookId(201l);

book.setTitle("Java How To Program");

*when*(bookRepository.save(book)).thenReturn(book);

//testing bookService.searchBook

Assert.*assertEquals*(book, bookService.addBook(book));

}

//@Test

**public** **void** bookServiceFetchAllBookTest() {

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

Book book1=**new** Book();

book1.setBookId(201l);

book1.setTitle("Spring In Action");

Book book2=**new** Book();

book2.setBookId(202l);

book2.setTitle("Java How To Program");

bookList.add(book1);

bookList.add(book2);

*when*(bookRepository.findAll()).thenReturn(bookList);

//testing bookService.fetchAllBook

Assert.*assertEquals*(bookList, bookService.fetchAllBook());

}

//@Test

**public** **void** subjectServiceFetchAllSubjectTest() {

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

Subject subjectObject = **new** Subject();

subjectObject.setSubjectId(1l);

subjectObject.setSubTitle("Bengali");

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

subject.setSubjectId(1l);

subject.setSubTitle("Computer");

subjectList.add(subjectObject);

subjectList.add(subject);

*when*(subjectRepository.findAll()).thenReturn(subjectList);

//testing subjectService.fetchAllSubject

//Assert.assertEquals(subjectList, subjectService.fetchAllSubject());

Assert.*assertNotSame*(subjectList, subjectService.fetchAllSubject());

}

//@Test

**public** **void** subjectServiceAddSubjectTest() {

Subject subjectObject = **new** Subject();

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

subject.setSubjectId(1l);

subject.setSubTitle("Computer");

*when*(subjectRepository.save(subject)).thenReturn(subject);

//testing subjectService.addSubject

Assert.*assertEquals*(subjectObject, subjectService.addSubject(subject));

}

//@Test

**public** **void** subjectServiceSearchSubjectTest() {

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

subject.setSubjectId(2l);

Optional<Subject> optionalsubject = Optional.*ofNullable*(subject);

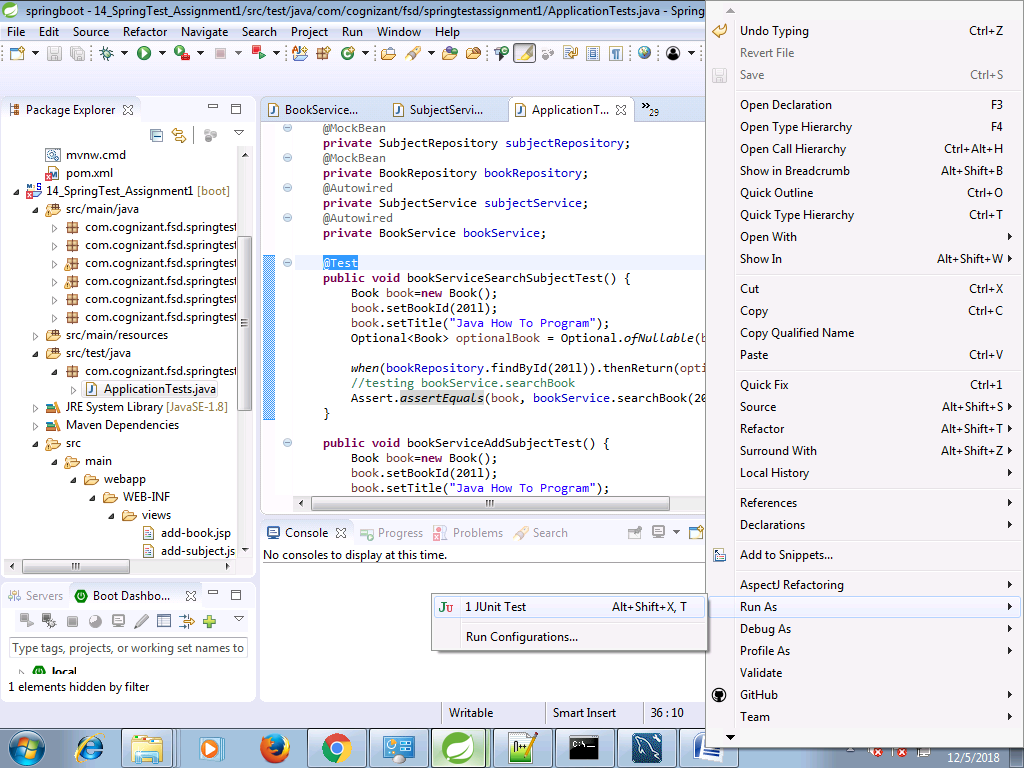
*when*(subjectRepository.findById(1l)).thenReturn(optionalsubject);

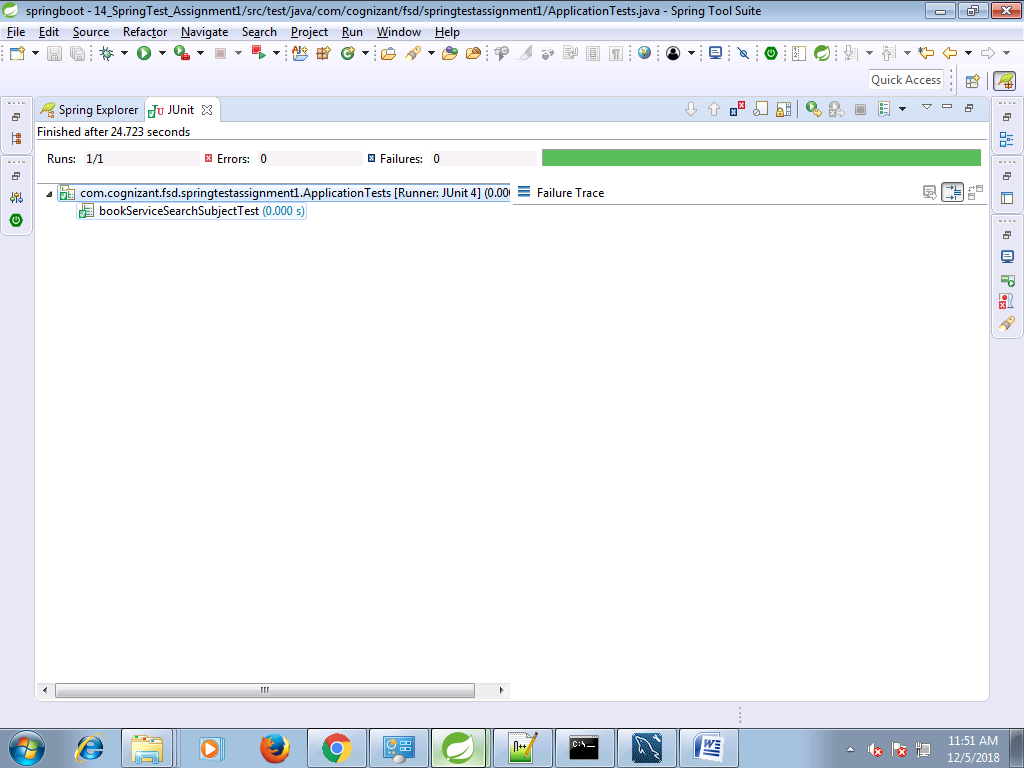
//testing subjectService.searchSubject

Assert.*assertEquals*(1l, subjectService.searchSubject(1l).getSubjectId().longValue());

}

}

****

****

**negative testing**

@Test

**public** **void** bookServiceSearchSubjectTest() {

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

book.setBookId(201l);

book.setTitle("Java How To Program");

Optional<Book> optionalBook = Optional.*ofNullable*(book);

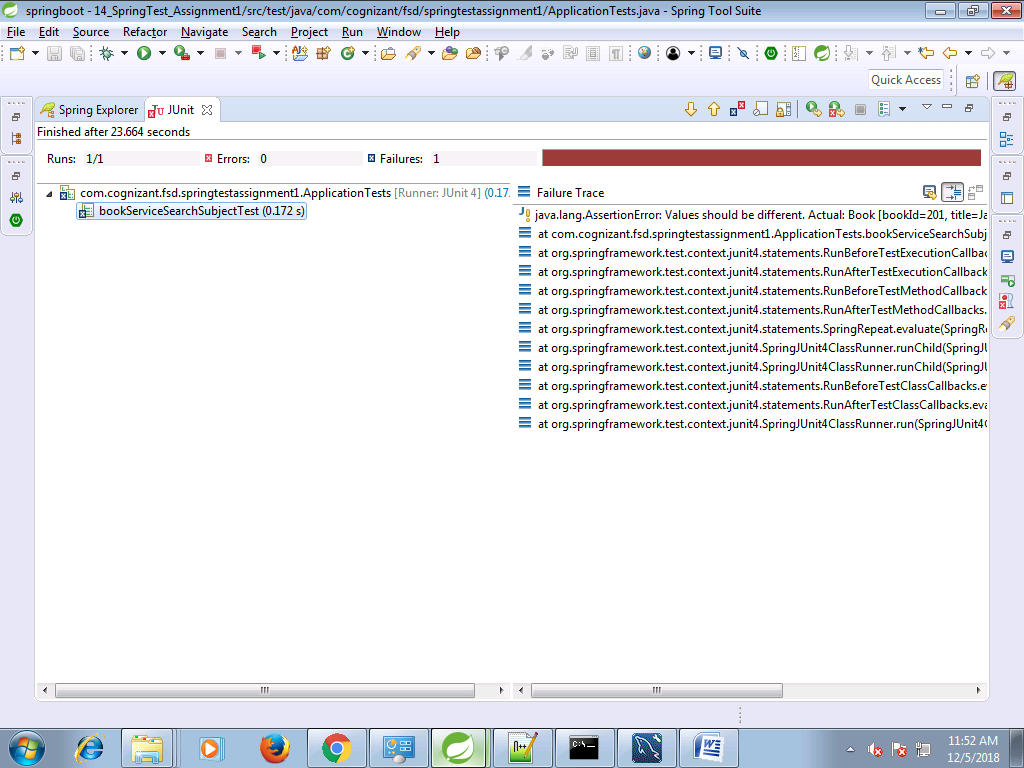
*when*(bookRepository.findById(201l)).thenReturn(optionalBook);

//testing bookService.searchBook

//Assert.assertEquals(book, bookService.searchBook(201l));

Assert.*assertNotEquals*(book, bookService.searchBook(201l));

}

****

@Test

**public** **void** bookServiceAddSubjectTest() {

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

book.setBookId(201l);

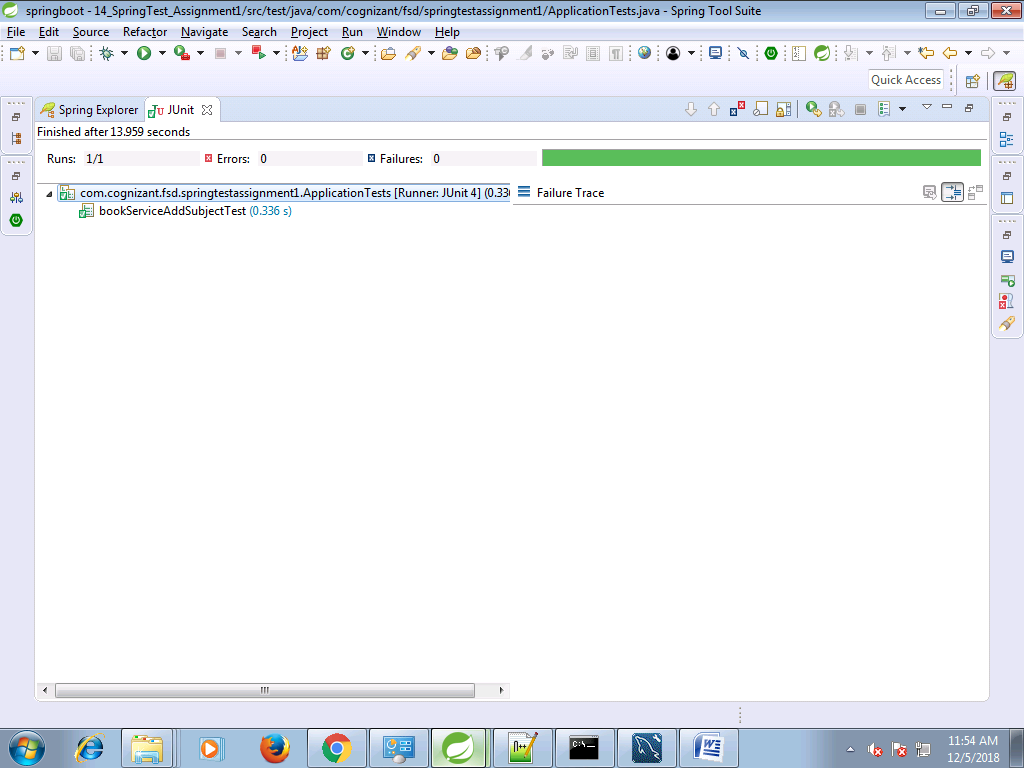
book.setTitle("Java How To Program");

*when*(bookRepository.save(book)).thenReturn(book);

//testing bookService.searchBook

Assert.*assertEquals*(book, bookService.addBook(book));

}

****

@Test

**public** **void** bookServiceAddSubjectTest() {

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

book.setBookId(201l);

book.setTitle("Java How To Program");

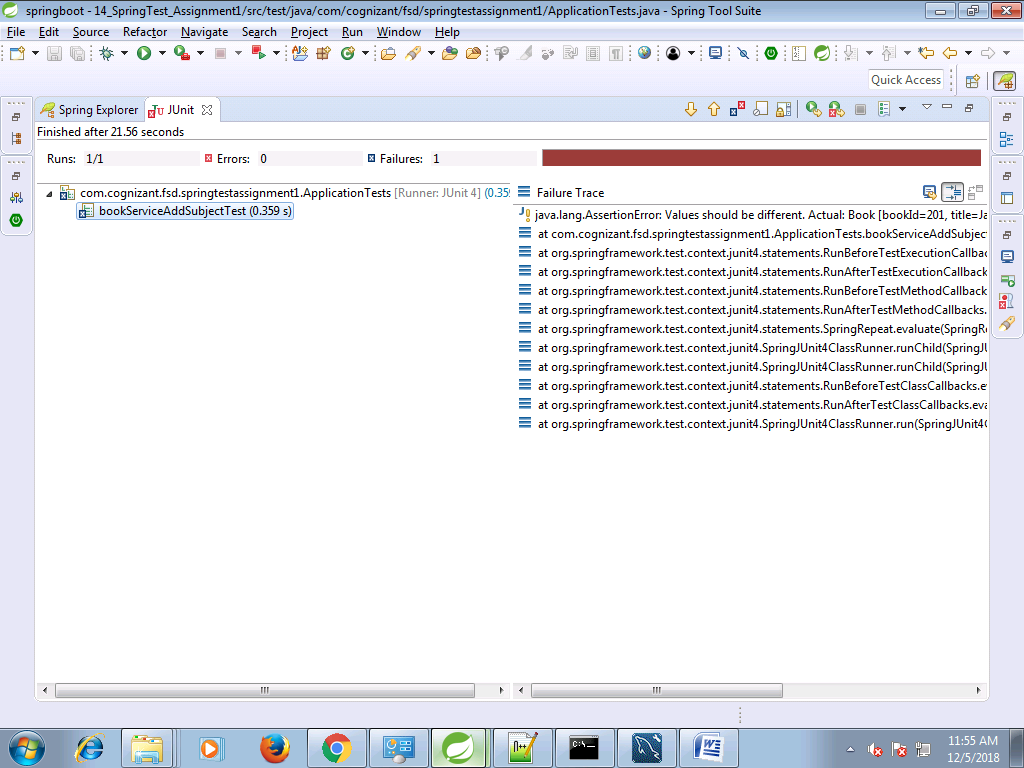
*when*(bookRepository.save(book)).thenReturn(book);

//testing bookService.searchBook

//Assert.assertEquals(book, bookService.addBook(book));

Assert.*assertNotEquals*(book, bookService.addBook(book));

}

****

@Test

**public** **void** bookServiceFetchAllBookTest() {

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

Book book1=**new** Book();

book1.setBookId(201l);

book1.setTitle("Spring In Action");

Book book2=**new** Book();

book2.setBookId(202l);

book2.setTitle("Java How To Program");

bookList.add(book1);

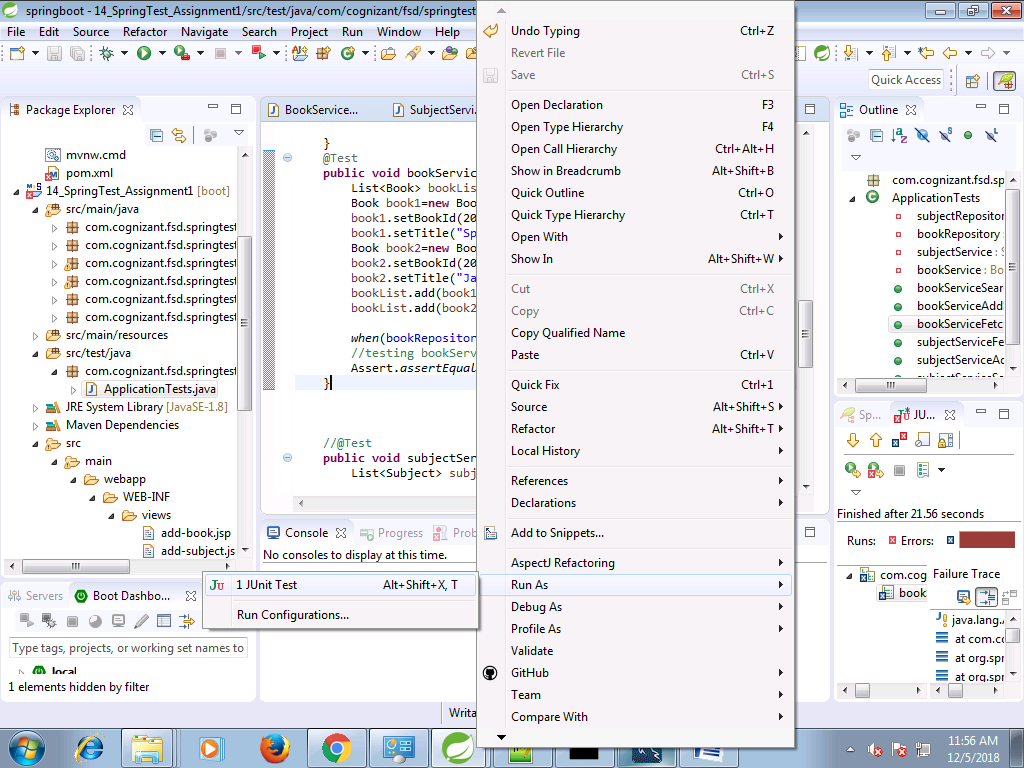
bookList.add(book2);

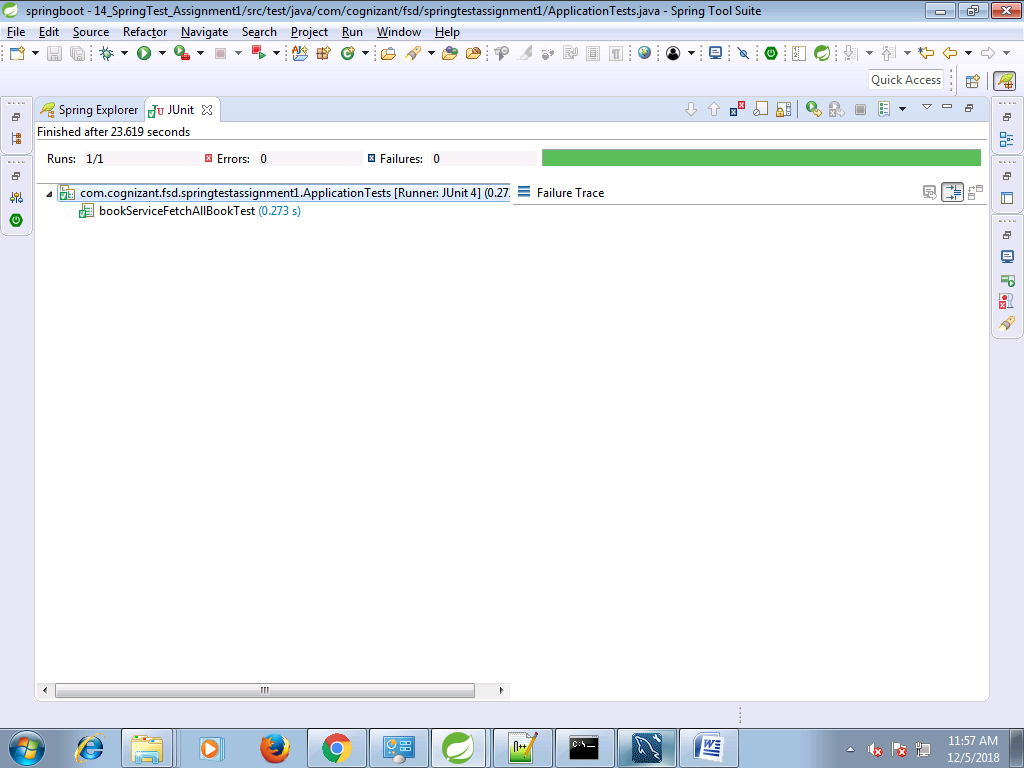
*when*(bookRepository.findAll()).thenReturn(bookList);

//testing bookService.fetchAllBook

Assert.*assertEquals*(bookList, bookService.fetchAllBook());

}

****

****

**negative testing**

@Test

**public** **void** bookServiceFetchAllBookTest() {

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

Book book1=**new** Book();

book1.setBookId(201l);

book1.setTitle("Spring In Action");

Book book2=**new** Book();

book2.setBookId(202l);

book2.setTitle("Java How To Program");

bookList.add(book1);

bookList.add(book2);

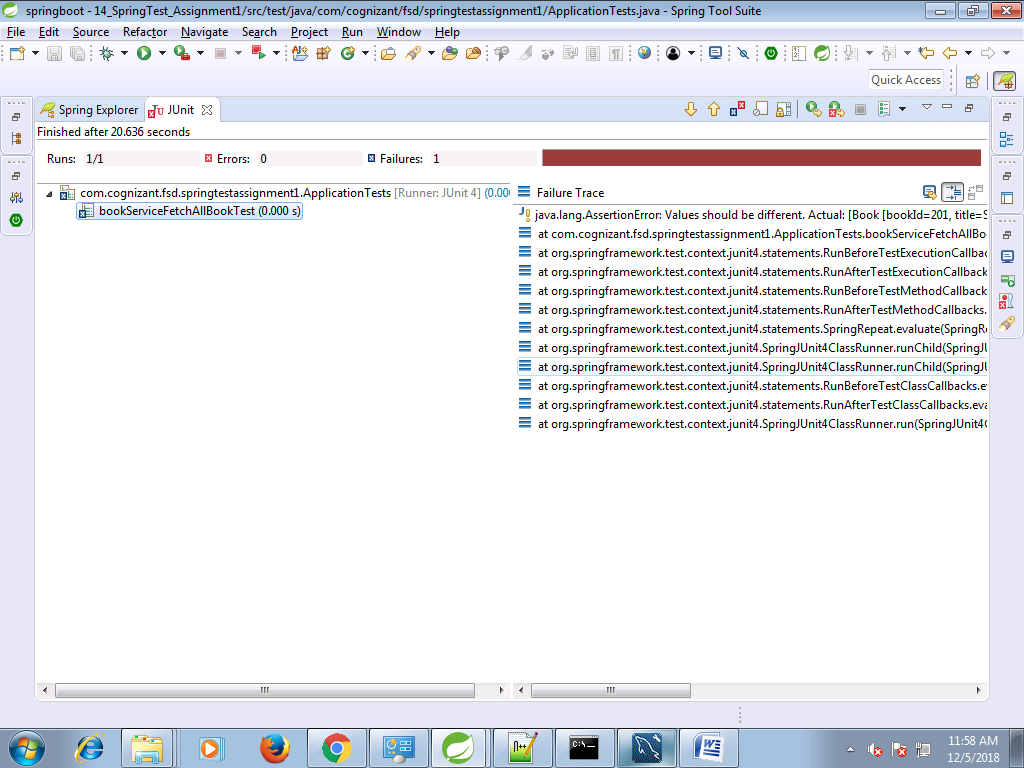
*when*(bookRepository.findAll()).thenReturn(bookList);

//testing bookService.fetchAllBook

//Assert.assertEquals(bookList, bookService.fetchAllBook());

Assert.*assertNotEquals*(bookList, bookService.fetchAllBook());

}

****

com.cognizant.fsd.springtestassignment1.repository.AbstractDao

package com.cognizant.fsd.springtestassignment1.repository;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.springframework.beans.factory.annotation.Autowired;

public abstract class AbstractDao {

@Autowired

private SessionFactory sessionFactory;

protected Session getSession() {

return sessionFactory.getCurrentSession();

}

public void persist(Object entity) {

getSession().saveOrUpdate(entity);

}

public void delete(Object entity) {

getSession().delete(entity);

}

}

com.cognizant.fsd.springtestassignment1.repository.BookRepository

**package** com.cognizant.fsd.springtestassignment1.repository;

**import** org.springframework.data.repository.CrudRepository;

**import** com.cognizant.fsd.springtestassignment1.model.Book;

**public** **interface** BookRepository **extends** CrudRepository<Book, Long>{

}

com.cognizant.fsd.springtestassignment1.repository.SubjectRepository

**package** com.cognizant.fsd.springtestassignment1.repository;

**import** org.springframework.data.repository.CrudRepository;

**import** com.cognizant.fsd.springtestassignment1.model.Subject;

**public** **interface** SubjectRepository **extends** CrudRepository<Subject, Long>{

}