System Requirements Specification

Index

For

Laptop Store App

Version 1.0

TABLE OF CONTENTS

Table of Contents

LAPTOP STORE APP

System Requirements Specification

SPRING BOOT RESTFUL APPLICATION

1 PROJECT ABSTRACT

The **Laptop Store App** is an application with a backend implemented using Spring Boot with a MySQL database. The application aims to provide a comprehensive platform for managing and organizing laptop related specifications.

The **Laptop Store App** project presents developers with a vital task: to design and implement a comprehensive set of test cases using Junit&Mockito to validate the functionality of the application.
Your task is to develop a robust suite of test cases that thoroughly evaluate the laptop store activities under various scenarios, ensuring accurate results and error-free performance.
The test suite aims to ensure the accuracy and reliability of the system, providing confidence in its performance and enhancing customer satisfaction.

Following is the requirement specifications:

	Laptop Store App
Modules	
1	Laptop
Event Module Functionalities	
1	Add new Laptop details
2	Update the existing Laptop details
3	Get the Laptop details by Id
4	Get all Laptops details
5	Delete a Laptop
6	Search for Laptop by name, price and band

2 REST ENDPOINTS

Rest End-points exposed in the controller along with method details:

2.1 LAPTOP CONTROLLER

URL Exposed		Purpose
1. /laptops		Fetches all the laptops
Http Method	GET	
Parameter	-	
Return	List<Laptops>	
2. /laptops		Add a new laptop details
Http Method	POST	
Parameter 1	Laptop	
Return	Laptop	
3. /laptops/{id}		Delete laptop with given laptop id
Http Method	DELETE	
Parameter 1	Long (id)	
Return	-	
4. /laptops/{id}		Fetches the laptop with the given id
Http Method	GET	
Parameter 1	Long (id)	
Return	Laptop	
5. /laptops/{id}		Updates existing Laptop info
Http Method	PUT	
Parameter 1	Long (id)	
Parameter 2	Laptop	
Return	Laptop	
6. /laptops/search?name={name}		Search the laptop with the given name
Http Method	GET	
Parameter 1	String (name)	
Return	List<Laptops>	
7. /laptops/search?price={price}		Search the laptop with the given price
Http Method	GET	
Parameter 1	Double (price)	
Return	List<Laptops>	

8. /laptops/search?brand={brand}		Search the laptop with the given brand
Http Method	GET	
Parameter 1	String (brand)	
Return	List<Laptops>	

3 TEMPLATE CODE STRUCTURE

PACKAGE: com.example.laptopstore

Resources

LaptopApplication (Class)	This is the Spring Boot starter class of the application.	Already Implemented

3.1 PACKAGE: com.example.laptopstore.test

Resources

Class/Interface	Description	Status
LaptopStoreTests	➔This class needs to contains Junit & Mockito test cases to verify the correctness of the methods in the **LaptopController** and **LaptopServiceImpl** classes ➔Make sure the test cases you write achieves 100% code coverage	Not Implemented

3.2 PACKAGE: com.example.laptopstore.repository

Resources

Class/Interface	Description	Status
LaptopRepository (interface)	• Repository interface exposing CRUD functionality for Laptop Entity. Do not modify, add or delete any method.	Already Implemented.

3.3 PACKAGE: com.example.laptopstore.service

Resources

Class/Interface	Description	Status
LaptopService (interface)	• Interface to expose method signatures for laptop related functionality. • Do not modify, add or delete any method.	Already implemented.

3.4 PACKAGE: com.example.laptopstore.service.impl

Class/Interface	Description	Status
LaptopServiceImpl (class)	• Implements LaptopService. • Contains template method implementation. • Need to provide implementation for laptop related functionalities. • Do not modify, add or delete any method signature	Already implemented.

3.5 PACKAGE: com.example.laptopstore.controller

Resources

Class/Interface	Description	Status
LaptopController (Class)	• Controller class to expose all rest-endpoints for laptop related activities. • May also contain local exception handler methods	Already implemented.

3.6 PACKAGE: com.example.laptopstore.dto

Resources

Class/Interface	Description	Status
LaptopDTO (Class)		Already implemented.

3.7 PACKAGE: com.example.laptopstore.entity

Resources

Class/Interface	Description	Status
Laptop (Class)		Already implemented.

3.8 PACKAGE: com.example.laptopstore.exception

Resources

Class/Interface	Description	Status
ResourceNotFoundException (Class)	• Custom Exception to be thrown when trying to fetch or delete the laptop info which does not exist.	Already implemented.
ExceptionHandlerController (Class)	• RestControllerAdvice Class for defining global exception handlers. • Contains Exception Handler for **InvalidDataException** class.	Already implemented.

4. EXECUTION STEPS TO FOLLOWED:

1. All actions like build, compile, running application, running test cases will be through Command Terminal.
2. To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top) → Terminal →New Terminal.
3. This editor Auto Saves the code.
4. If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.
5. These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.
6. To execute and run test cases:
   mvn clean install exec:java -Dexec.mainClass="com.example.laptopstore.LaptopApplication" -DskipTests=true
7. You need to use **CTRL+Shift+B** - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.