System Requirements Specification Index

For

Grocery App- Junit + Mockito

Version 1.0

IIHT Pvt. Ltd.
fullstack@iiht.com

TABLE OF CONTENTS

Table of Contents

GROCERY APP
System Requirements Specification

1 PROJECT ABSTRACT

The **Grocery 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 Grocery App.

Your task is to develop a robust suite of test cases that thoroughly evaluate the grocery manager's basic activities under various scenarios, ensuring accurate results and error-free performance.

The **Grocery App** test suite aims to ensure the accuracy and reliability of the system, providing confidence in its performance and enhancing customer satisfaction.

2 CODE STRUCTURE

2.1 PACKAGE: COM.GROCERYAPP.SERVICE

Resources

Class/Interface	Description	Status
GroceryManager(class)	• This class represents a service for adding and removing grocery items from collection. • Methods takes the groceryItem as input parameters and performs addition and removal of same from collection. • These methods use validation methods from ValidationUtils class in validation package.	Already implemented.

	• Don't modify any methods in this class as this is already implemented.	

2.2 PACKAGE: COM.GROCERYAPP.VALIDATION

Resources

Class/Interface	Description	Status
ValidationUtils(class)	• This class represents validation methods.	Already implemented.
	• Don't modify any methods in this class as this is already implemented.	

2.3 PACKAGE: COM.GROCERYAPP.TEST

Resources

Class/Interface	Description	Status
GroceryAppTest(class)	• This class contains JUnit test cases to verify the correctness of the methods in the GroceryManager and ValidationUtils class. • Make sure while writing the test cases for GroceryManager, mock the Validation methods of ValidationUtils. • Write Independent test cases for ValidationUtil methods. • Make sure the test cases you write achieves 100% code coverage.	To be implemented.

3 EXECUTION STEPS TO FOLLOW

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="mainapp.MyApp" -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.**