Test Plan for Product Class

1. Introduction

This document outlines the unit testing strategy for the `Product` class, ensuring all functionalities work as expected. Tests will be implemented using **JUnit 5**.

2. Objectives

- Verify correct initialization of `Product` objects.
- Test getters and setters for expected behavior.
- Ensure validation logic in `setProductCode` and `setProductName` works correctly.
- Validate string truncation in `productName`.
- Confirm `toString()` returns the expected format.

3. Scope

- **In-Scope:** Unit testing individual methods of the `Product` class.
- **Out-of-Scope:** Integration testing with `Utilities` class.

4. Test Approach

- Each method will be tested using multiple test cases.
- Boundary values will be included for `productCode` and `productName`.
- Assertions will compare expected vs. actual results.

5. Test Environment

- **Language:** Java 17+
- **Testing Framework:** JUnit 5
- **IDE:** IntelliJ IDEA / Eclipse
- **Tools:** Maven / Gradle (if applicable)

6. Test Cases Overview

ID	Test Case	Expected Result
TC-01	Create a `Product` object with valid values	Object is created correctly
TC-02	Test `getProductName()`	Returns expected name
TC-03	Test `getProductCode()`	Returns expected code
TC-04	Test `getUnitCost()`	Returns expected unit cost
TC-05	Test `isInCurrentProductLine()`	Returns expected boolean value
TC-06	Test `setProductCode()` with valid code	Updates successfully
TC-07	Test `setProductCode()` with invalid code	Does not update
TC-08	Test `setProductName()` with valid name	Updates successfully
TC-09	Test `setProductName()` with long name	Truncates to 20 characters
TC-10	Test `toString()` output	Matches expected format