SOFTWARE ENGINEERING HANDS-ON 4
TEAM 12
PES UNIVERSITY
AUGUST – DECEMBER 2022 SEMESTER 5

Problem Statement – 1: Unit Testing
• Discuss with your teammates and demarcate units in your code base

Each web page component can be tested individually as their functionality depends on the features and states of that respective page like

1. **The separate page for each food item where the value of the price is calculated based of the proportion size and the quantity can be tested individually as it is independent of the other paths**
2. **The payment gateway can similarly be tested individually**

• Develop test cases for both valid and invalid data

1. **We use API such as Stripe for our payment gateway which handle validity of the card details and other information related to the payment**
2. **The other calculation such as price and quantity are tested manually by entering values according to each test cases**

• Ideate how you could further modularize larger blocks of code into compact units with your teammates

1. **We modularise our codes by splitting each functionality of the website into separate components which can be reused accordingly eg the headers and footers**
2. **We use dynamic routing for each item so that we can use the same web design code to render the various items**

Problem Statement – 2: Dynamic Testing
Dynamic testing involves execution of your code to analyze errors found during execution. Some common techniques are Boundary Value Analysis and Mutation Testing.

Problem Statement – 2.a: Boundary Value Analysis
When it comes to finding errors in your code base, they are often found at locations where a condition is being tested. Due to this, developers often use Boundary Value tests to reduce defect density.
• How would you define a boundary test?

1. **We intend to use dynamic testing in our web application during login and payment gateway**
2. **If you attempt to perform static analysis of a web application that requires login or payment, Application Catalog will not be able to test any of the pages other than the login page.**

• Build your boundary test cases and execute them

1. **Some of our test cases include making sure the quantity of products never go below zero**
2. **Stripe takes care of veriffying the bank details and the payment processing**

Problem Statement – 2.b: Mutation Testing
• Using your isolated units from the first problem statement,ideate with your team mates on how to mutate the code

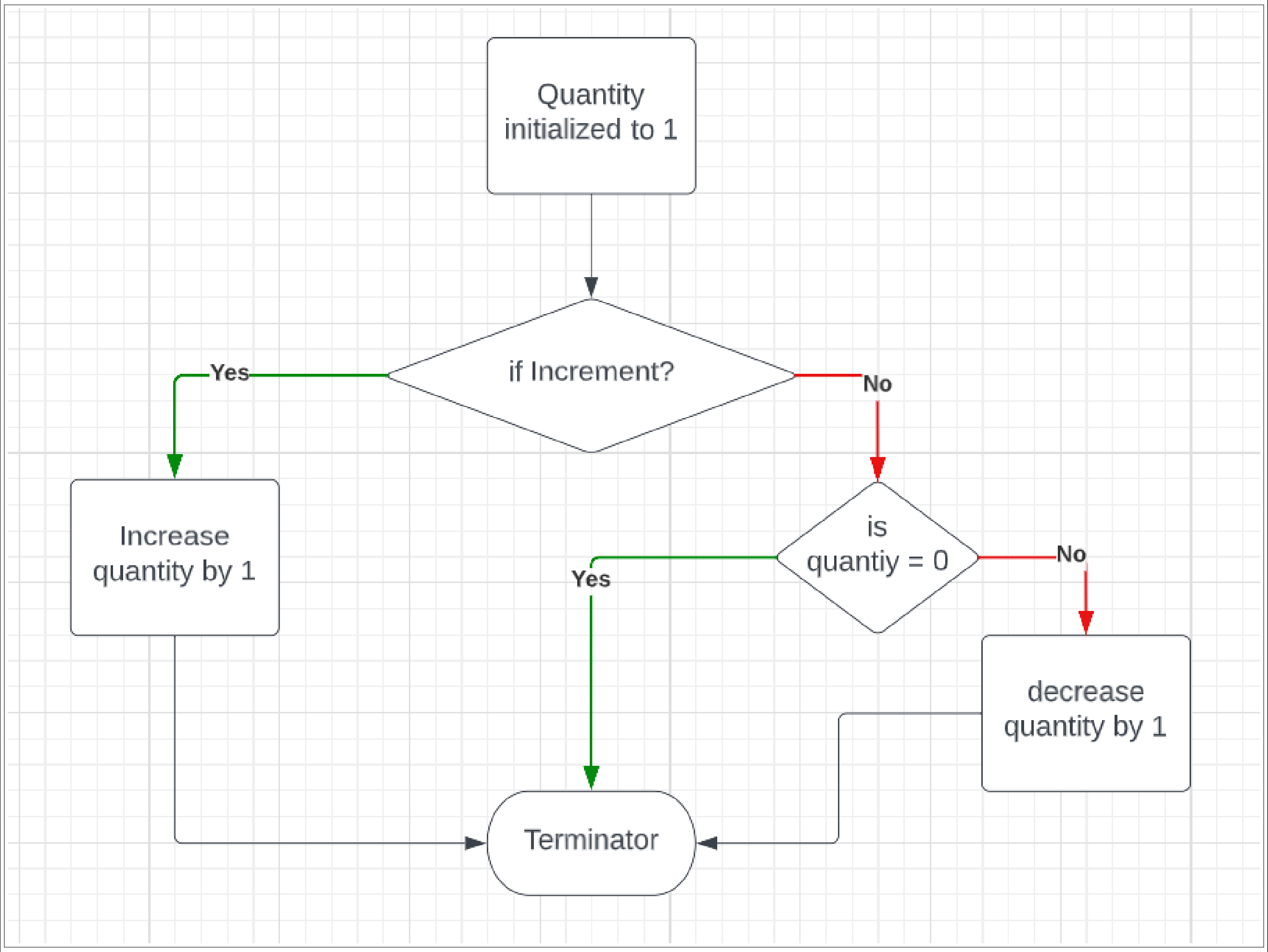
- **Testing individual web software component in isolation cannot detect interaction faults, which occur in communication among web software components. Improperly implementing and testing the communications among web software components is a major source of faults. a novel solution to the problem of integration testing of web applications by using mutation analysis, "web mutation testing."**
- **We intend to do so by reversing the operations of the buttons which are responsible for increasing and decreasing the quantity ordered**

• Develop at least 3 mutants of the functioning code and test all 4 code bases using the test case from the first problem statement

1. **MUTANT 1: The price values is independent of the portion size**
2. **MUTANT 2: The price values is independent of the quantity ordered**
3. **MUTANT 3: The price values is not updated using setState function**
   We there by try to identify pieces of code that are not tested properly. To identify hidden defects that can't be detected using other testing methods.

Problem Statement – 3: Static Testing
Static testing involves validating your code without any execution. Under this problem statement, you will be expected to analyse and calculate the cyclomatic complexity of your code.

• Using the unit you selected in the first problem statement as an example, develop the control flow graph of your problem statement.



• Using the Control flow graph, calculate the cyclomatic complexity of your code.

Complexity = E – N + 2*P
= 7 – 6 + 2
= 3

• Using the cyclomatic complexity as an indicator, Ideate and code your unit again to reduce complexity

Here we cannot reduce it more as this is the maximum complexity attained by making sure all the condition are followed

Problem Statement – 4: Acceptance Testing

SOFTWARE ENGINEERING HANDS-ON 4
TEAM 12
PES UNIVERSITY
AUGUST – DECEMBER 2022 SEMESTER 5

Assume your neighbouring team is the client for your code. Give them an idea of what your product is and the software requirements for the product.

Our project is an easy and simple to use interface for food delivery system ,where we take care of placing orders and following it through until it is safely and timely delivered

• Exchange your code base and test each others projects to see if it meets user requirements
• If you identify a bug in the project you are testing, inform the opposing team of the bug
• As a team, based in clients experience, ideate modifications to the existing project that could improve client experience

1. **Acceptance testing is a technique performed to determine whether or not the system has met the requirement specifications**
2. **We would improve the UI design to be more intutional and make the delivery time minimal by improving the algorithm for job assignment**

Problem Statement – 5: Maintenance Activities
Once a product is completed, it is handed off to a service based company to ensure all maintenance activities are performed without the added expenditure of skilled developers. However, a few tasks are performed by the maintenance team to gauge the product better. In this problem statement, you will be asked to experiment with your code.
• Exchange code bases with your neighboring teams and reverse engineer a block of code in order to understand it's functionality

We intend to reverse engineer the block of code to peep into the working of the project and how each functions work together simultaneously

• After understanding the code block, Re-Engineer the code
o Ideate how to refactor the code and the portion of the code base you would have to change
o Discuss how the new changes would impact the time and space complexity of the project during execution

Re engineering is done to improve on the current implementation of the software .
We intend to refactor our code by

- **Getting rid of switch statements**
- **Making sure the conditionals are descriptive**
- **Avoiding code redundancies**
- **Assigning one functionality to one function**

The new changes would reduce the time and space complexity which thereby lets the website render faster and gives the user a smoother experience

• After Reverse Engineering and Re-Engineering the code, perform acceptance testing between the teams

- **Acceptance testing is done based on the functional testing of user story by the team during the implementation phase.**
- **It helps demonstrate that required business functions are operating in a manner suited to the real-world circumstances and usage.**