Object Oriented Structure Measurement of SPL I

for

Shop Assistant

Submitted To

Dipok Chandra Das
Assistant Professor
Institute of Information Technology (IIT)
Noakhali Science and Technology University

Submitted by

Prosanto Deb (ASH1925005M)
Sultana Marjan (BKH1925010F)
Shahriar Ahmed (ASH1925015M)
Md. Alamgir Hossain (ASH1925016M)
Ratno Kumer Tripura (ASH1825042M)

Course Title: Software Metrics Lab

Course Code: SE 3204

Date of Submission: 05/03/2023

Table of Contents

Table of Contents ii

1. Project Information 1

2. Depth of Inheritance Tree (DIT) 1

3. Coupling Between Objects 2

4. Afferent Coupling (Ca) 3

5. Efferent Coupling(Ce) 4

6. Number of Children (NOC) 4

7. Response for a Class (RFC) 5

Table of Figures

Figure 1 Depth Of Inheritance Tree 2
Figure 2 Coupling Between Objects 3
Figure 3 Afferent Coupling 3
Figure 4 Efferent Coupling 3
Figure 5 Number of Child (NOC) 3
Figure 6 Response for a class (RFC) 3

1. Project Information

Project Name	**Shop Assistant**
Supervised By	**Dipanita Saha** Assistant Professor Institute of Information Technology(IIT) Noakhali Science and Technology University
Team Members	**Prosanto Deb** (ASH1925005M)
	Sultana Marjan (BKH1925010F)
	Shahriar Ahmed (ASH1925015M)
	Md Alamgir Hossain (ASH1925016M)
	Ratno Kumer Tripura (ASH1825042M)
GitHub Link	Shop Assistant
Project Report	Shop Assistant Report

2. Depth of Inheritance Tree (DIT)

Metric	**Depth of Inheritance Tree (DIT)**
Definition	Depth of inheritance of the class is the DIT metric for the class. DIT will be the maximum length from the node to the root of the tree. It is a measure of how many ancestor classes can potentially affect this class.
Viewpoints	• The deeper a class is in the hierarchy, the greater the number of methods it is likely to inherit, making it more complex to predict its behavior. • Deeper trees constitute greater design complexity, since more methods and classes are involved. • The deeper a particular class is in the hierarchy, the greater the potential reuse of inherited methods.

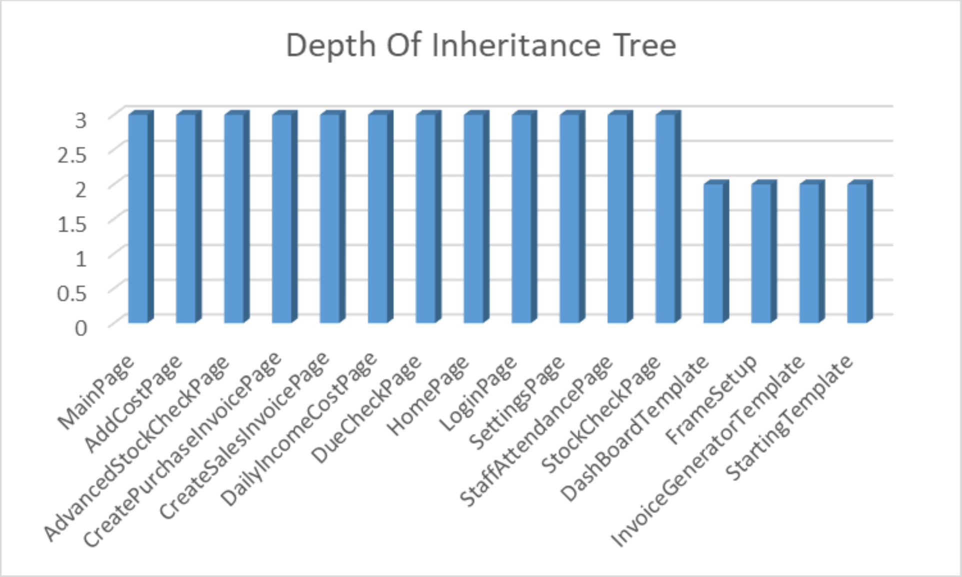

Figure 1 Depth Of Inheritance Tree

3. Coupling Between Objects

Metric	**Coupling Between Object (CBO)**
Definition	Coupling Between Objects (CBO) is a software metric that measures the level of interdependence between two or more classes or objects in a software system. In other words, CBO indicates the number of other classes and the degree to which those classes are dependent on a given class or object.
Viewpoints	• High CBO indicates tight coupling between classes, which can make the code more complex and harder to maintain. • Low CBO indicates loose coupling, which can make the code more modular and easier to modify. • CBO can be reduced by promoting good design principles such as encapsulation, abstraction, and information hiding.

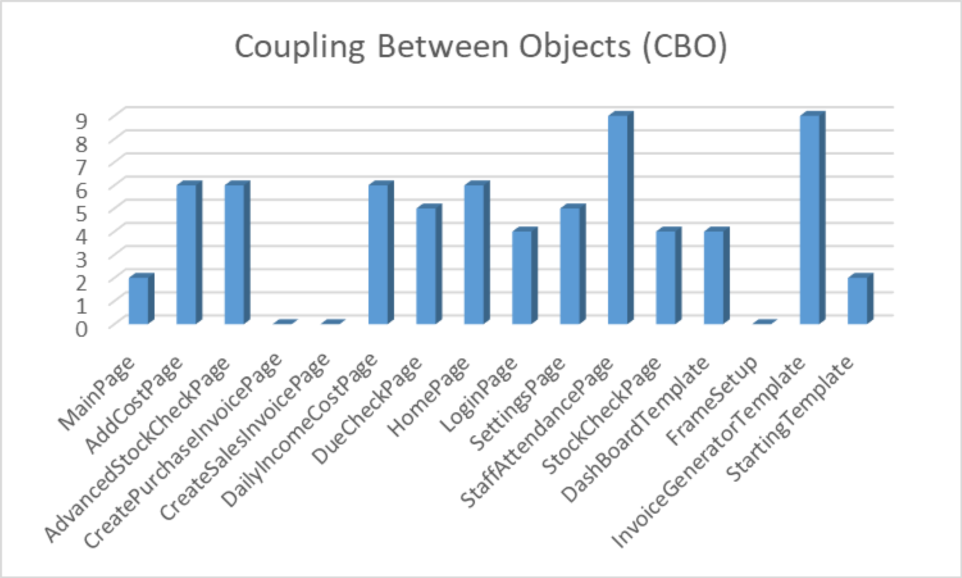

Figure 2 Coupling Between Objects

4. Afferent Coupling (Ca)

Metric	**Afferent coupling (Ca)**
Definition	Afferent coupling (Ca) is the number of classes from other packages that depends on the classes within the subject package. $I = \frac{C_e}{C_a + C_e}$ Here, I is instability metric

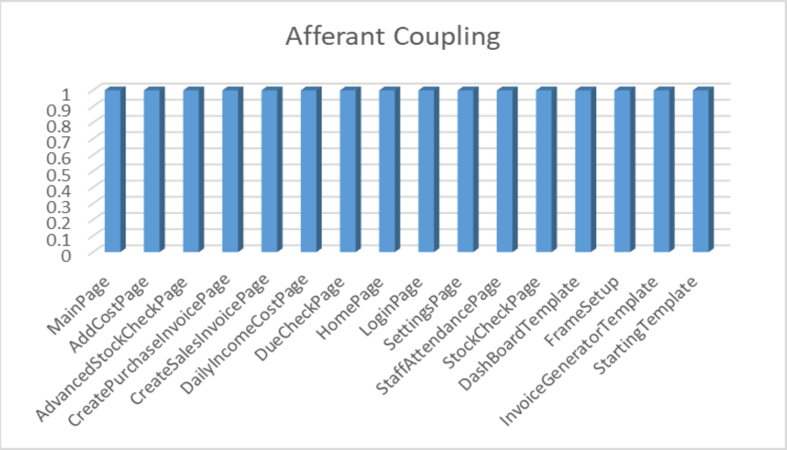

Figure 3 Afferent Coupling

5. Efferent Coupling(Ce)

Metric	**Efferent coupling (Ce)**
Definition	Efferent coupling (Ce) is the number of classes in other packages that the classes in the subject package depend on" via class relationships. $$I = \frac{C_e}{C_a + C_e}$$ Here, I is instability metric

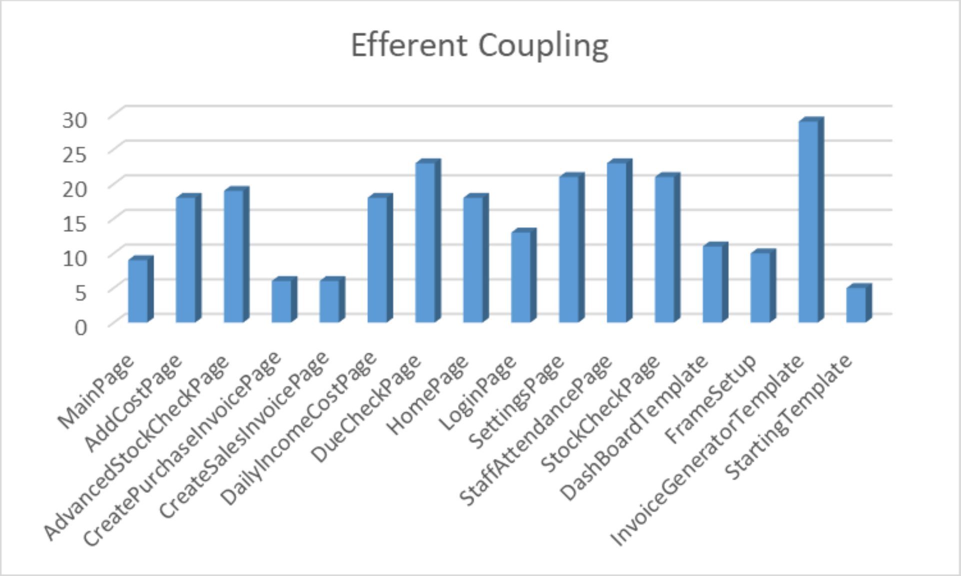

Figure 4 Efferent Coupling

6. Number of Children (NOC)

Metric	**Number of Children (NOC)**
Definition	NOC is the number of immediate subclasses subordinated to a class in the class hierarchy.
Viewpoints	• Greater the number of children, greater the reuse, since

	inheritance is a form of reuse. • If a class has a large number of children, it may be a case of misuse of sub classing. • If a class has a large number of children, it may require more testing of the methods in that class.

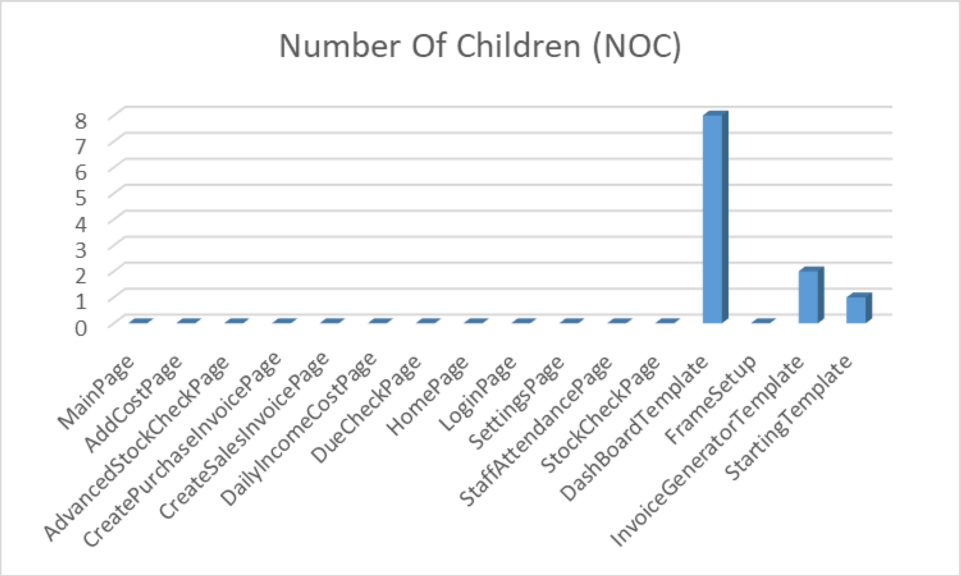

Figure 5 Number of Children (NOC)

7. Response for a Class (RFC)

Metric	**Response For a Class (RFC)**
Definition	RFC = RS (where RS is the response set for the class) The response set for the class can be expressed as- RS = {M} Uall i {Ri} where { R,} = set of methods called by method i and { M} = set of all methods in the class.
Viewpoints	• If a large number of methods can be invoked in response to a message, the testing and debugging of the class becomes more complicated • The larger the number of methods that can be invoked from a

	class, the greater the complexity of the class. • A worst-case value for possible responses will assist in appropriate allocation of testing time.

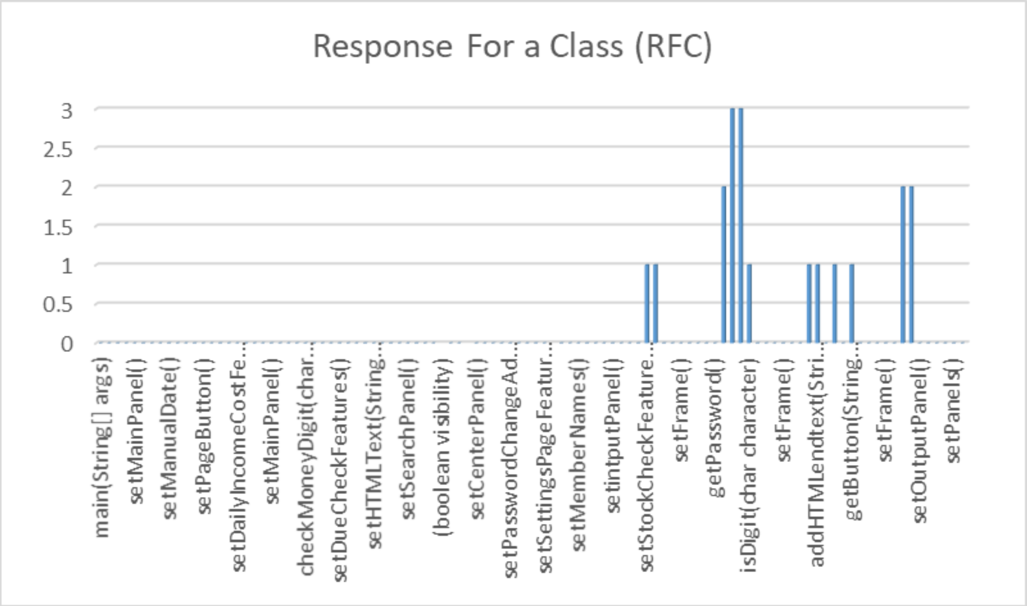

Figure 6 Response for a class (RFC)