EXPERIMENT NO 7

Aim: Design test cases and perform white box testing. (Evaluate code and the internal structure of software.)

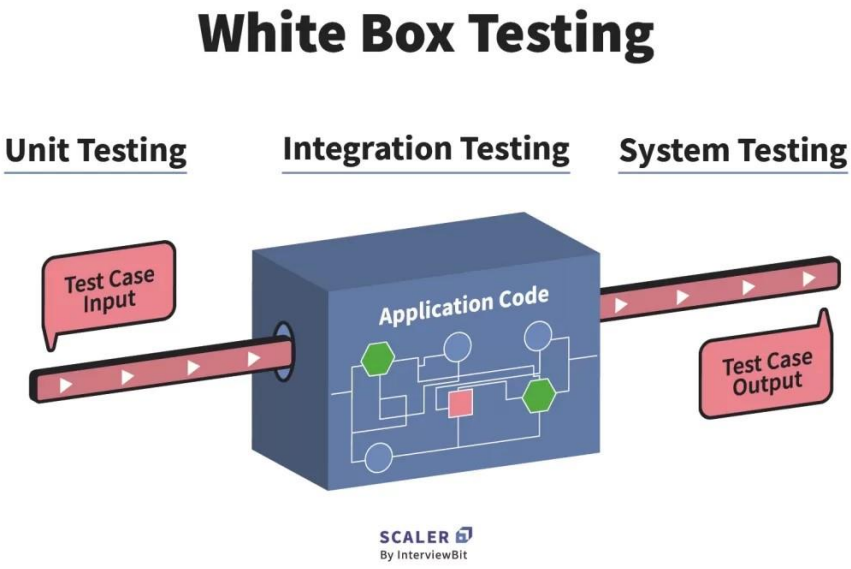
Theory:

The box testing approach of software testing consists of black box testing and white box testing. We are discussing here white box testing which is also known as glass box testing, structural testing, clear box testing, open box testing, and transparent box testing. It tests the internal coding and infrastructure of a software focus on checking predefined inputs against expected and desired outputs. It is based on the inner workings of an application and revolves around internal structure testing. In this type of testing programming skills are required to design test cases. The primary goal of white box testing is to focus on the flow of inputs and outputs through the software and strengthen the security of the software.

The term 'white box' is used because of the internal perspective of the system. The clear box or white box or transparent box name denotes the ability to see through the software's outer shell into its inner workings.

Developers do white box testing. In this, the developer will test every line of the code of the program. The developers perform the White-box testing and then send the application or the software to the testing team, where they will perform the black-box testing, verify the application along with the requirements, identify the bugs, and send it to the developer.

The developer fixes the bugs and does one round of white box testing and sends it to the testing team. Here, fixing the bugs implies that the bug is deleted, and the particular feature is working fine on the application.



Control Flow Software Testing

Is a form of white-box testing that focuses on the flow of control within a software program. It ensures that all the paths in the software code are covered during testing, which helps in evaluating the accuracy and efficiency of the program's structure. This type of testing checks how the control flows through the program and whether it follows the intended execution paths.

1. **Control Flow Graph (CFG)**: A Control Flow Graph represents the flow of the program using nodes and edges, where:
    - **Nodes** represent the statements or instructions in the code.
    - **Edges** represent the flow of control from one statement to another.
2. **Basic Elements**:
    - **Decision Nodes**: Points in the code where the control splits, such as if-else statements or loops (for, while).
    - **Sequence Nodes**: Simple sequence of instructions without branching.
    - **Loop Nodes**: Represent repetitive structures, indicating loops or iterations.
3. **Objectives**:
    - To ensure that all decision points and paths in the code are tested.
    - To identify unreachable code, redundant statements, and areas where the control flow might cause unexpected behavior.
    - To verify that all loops and conditional branches operate as expected under various conditions.

Techniques Used:

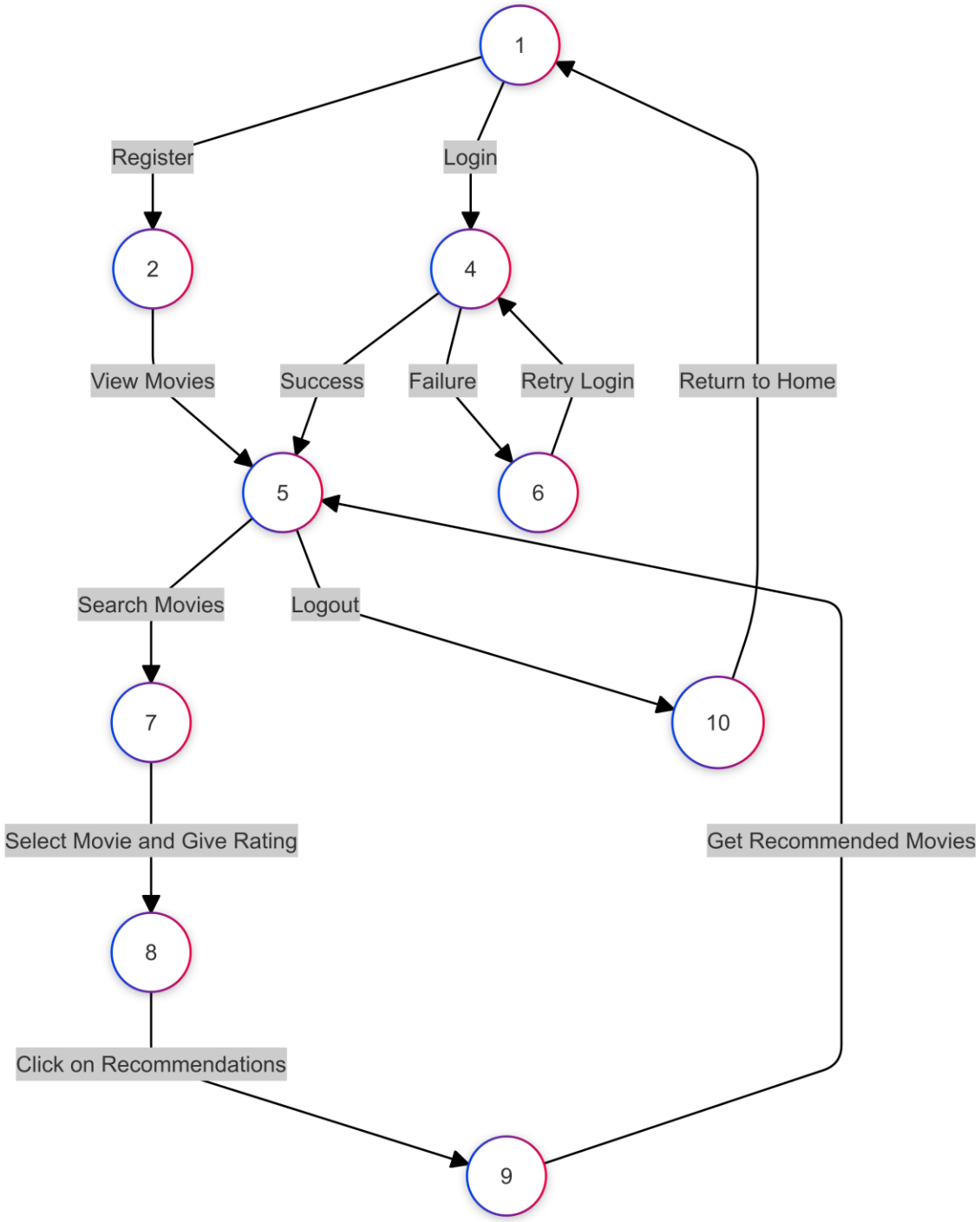
1. **Statement Coverage**: This ensures that every executable statement in the code is executed at least once.
2. **Branch Coverage**: Ensures that every branch (true and false conditions of decisions) is executed.
3. **Path Coverage**: Involves testing all possible paths within the program, ensuring that every unique path is executed.
4. **Loop Testing**: Ensures that the loops run for zero, one, and multiple iterations, covering all corner cases.

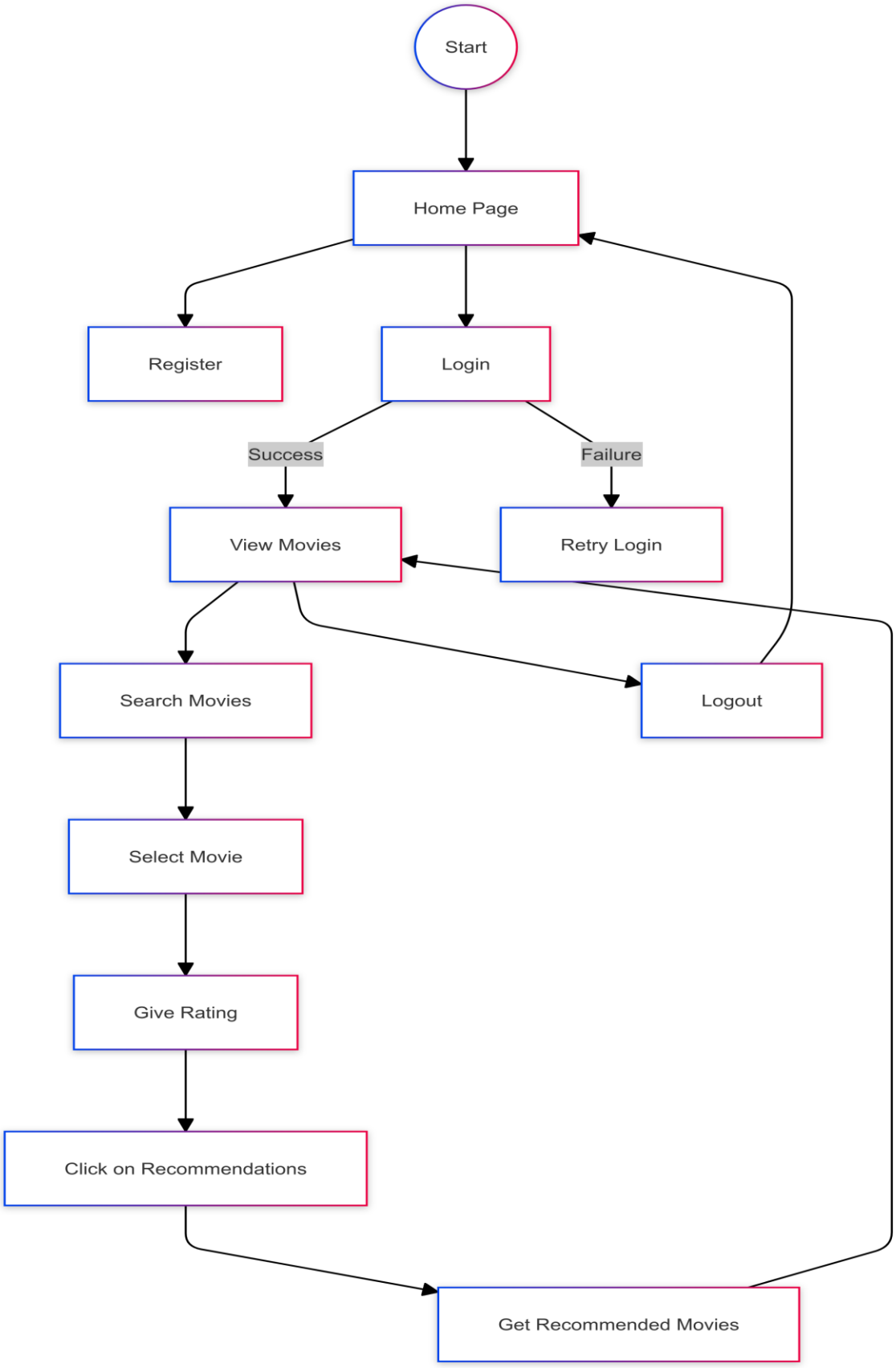
CONCLUSION:

In conclusion, white-box testing is a critical step in software development, evaluating the code's internal structure. It ensures code quality, reduces risks, improves security, and enhances performance. This method, when integrated into a comprehensive testing strategy, helps deliver reliable and secure applications.

OUTPUT:

CFG:


1
Register
Login
2
4
View Movies
Success
Failure
Retry Login
Return to Home
5
6
Search Movies
Logout
7
10
Select Movie and Give Rating
Get Recommended Movies
8
Click on Recommendations
9


Start
Home Page
Register
Login
Success
Failure
View Movies
Retry Login
Search Movies
Logout
Select Movie
Give Rating
Click on Recommendations
Get Recommended Movies

CALCULATION:

Cyclomatic complexity:

1. V(G) = edges - nodes + 2p

   Where,

   P = number of unconnected parts of the graph = 1

   ∴ V(G) = 12 - 10 + 2

   ∴ V(G) = 4

2. V(G) = number of predicate nodes + 1.

   A predicate node is a node with more than one edge emanating from it.

   ∴ V(G) = 3 + 1 = 4

3. V(G) = number of regions in the control flow graph.

   ∴ V(G) = 4