# Factory Simulation Software System

## Test Results

**Vedant Pate** **14CS30021**

**Shubham Kumar** **14CS30033**

# Table of Contents

## BLACK BOX TESTING:-

It includes testing of all functions used in the system.Boundary Value analysis is done in this testing depending on the values of parameters and variables present in the system.

## BBT for Add_Machine

| Method Name | **Add_Machine** |
|---|---|
| Parameter | **Machine_Name,Quantity, Repair_Time,MTTF** |
| Machine_Name | Lathe Machine<br>(Error Message If machine Name already Exists) |
| Quantity | 50<br>(Error Message if Quantity is more than 1000) |
| MTTF(Mean Failure Time) | 90(in days) |
| Repair_Time(Time require to repair Machine) | 6(in days) |

After this method this machine will get added in the factory.

## BBT for Add_Adjuster

| Method Name | **Add_Adjuster** |
|---|---|
| Parameter | **Adjuster_Id,**<br>**Machine_list**(Machines in the factory that adjuster can repair)<br>**Quantity**(Number of Adjusters of this type) |
| Adjsuter_Id | Group_1 |
| Machine_List | Lathe_Machine, Turning_Machine |
| Quantity | 10<br>(Error Message if greater than 1000) |

After this method this adjuster group will be available to work in the factory.

# Test Cases:-

## Add Machine method-

### Equivalence Class:-

#### Class 1:Success Message

Machine Name-Lathe

Quantity-50

Repair Time-6

Output-Machine gets added in the factory.

**Machine Details**

| Machine Type | MTTF(Days) | Repair Time | Quantity |
|---|---|---|---|
| Lathe Machine | 90 | 6 | 50 |

#### Class 1:Error Message

Class :-quantity (>1000|| <1)

Machine Name-Lathe

Quantity:  -1111

Repair Time-6

Output- Error Message.

## Boundary Conditions:-

Case 1:-(Quantity < 1000 && quantity >0)

Machine Name-Lathe

Quantity-1000

Repair Time- 45

Output-Machine gets added in the factory.

**Machine Details**

| Machine Type | MTTF(Days) | Repair Time | Quantity |
|---|---|---|---|
| Lathe | 461 | 45 | 1000 |

Case 2:-(Repair Time>0)

Machine Name-Lathe

Quantity-1000

Repair Time- 0


Case 3:-(MTTF>0)

Machine Name-Lathe

Quantity-1000

MTTF- 0


## ADD_ADJUSTER Method:-

While Adding an Adjuster Quantity should be positive (quantity>0&& quantity <=1000),otherwise error message appears.

Boundary conditions can be checked for this method also.(quantity=0||quantity=1000)

Adjuster Id Can not Be Null.

# BBT for Get_Machine_Details

| Method Name | **Get_Machine_details** |
|---|---|
| Parameter | Machine_Name |
| Machine_Name | Lathe_Machine |
| Quantity | 50 |
| MTTF(Mean Failure Time) | 90(in days) |
| Repair_Time(Time require to repair Machine) | 6(in days) |
| Machine_Utilisation_in_ factory | 87.67% |

After this method machine details and its utilisation percentage obtained as shown in table.

## BBT for Get_AdJuster_Details

| Method Name | **Get_Adjuster_Details** |
|---|---|
| Parameter | Adjuster_Id |
| Adjsuter_Id | Group_1 |
| Machine_List | Lathe_Machine, Turning_Machine |
| Quantity | 10 |
| Adjuster_Utilisation_in_ Factory | 40.82% |

After this method adjuster_group details and its utilisation percentage obtained as shown in above table .

# BBT for Simulation

| Method Name | **Siumulation_Function** |
|---|---|
| Parameter | Machine_Array<br>Adjuster_Group_Array<br>Simulation_Time |
| Simulation TIme Period | 1 Year |
| Average_Adjuster_<br>Utilisation | Group_1 |
| Average_Machine_<br>Utilisation | Lathe_Machine,<br>Turning_Machine |

Average Machine and Average Adjuster utilisation percentage obtained by using this method.

| Field | Percentage Utilisation |
|---|---|

| | |
|---|---|
| Average Machine Utilisation Output | 86.95% |
| Average Adjuster Utilisation Output | 36.37% |

## BBT for Get_Machinewise_Utilisation

| | |
|---|---|
| Method Name | **Machine_Utilisation** |
| Parameter | No Parameter |
| Output Obtained | |

| Machine Name | % Utilisation |
|---|---|
| Lathe Machine | 87.67% |
| Turning | 83.21% |
| CNG | 94.52% |

## BBT for Get_Adjusterwise_Utilisation

| | |
|---|---|
| Method Name | **Adjuster_Utilisation** |
| Parameter | No Parameter |
| Output Obtained | |

| Adjuster Id | % Utilisation |
|---|---|

| | |
|---|---|
| | Group-1 \| 40.82%<br>Group-2 \| 21.91%<br>Group-3 \| 38.16% |

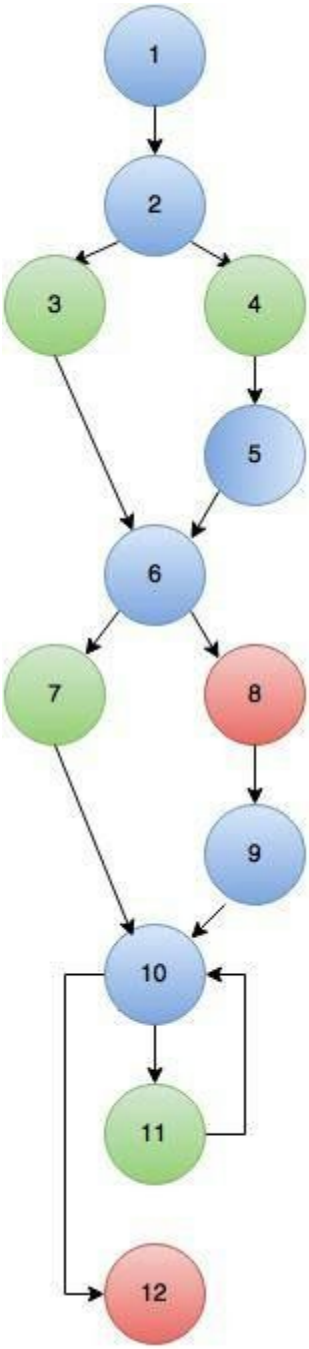
After this methods Machine and adjuster wise utilisation obtained respectively.

## White BOX TESTING:-

### ADD Adjuster Method:-

```
Void Add_adjuster(String Adjuster_id,Machine_type Machine_list,int
quantity)     // 1
{
        if(quantity>=1&&quantity<=1000)              // 2
        {
                Copy quantity                 // 3
        }
        else                                  // 4
        {
                Error message & Re-enter the quantity                 // 5
        }
        if(Adjuster_id!=NULL)                 // 6
        {
                Copy id                 // 7
        }
```

```
        else                                    // 8
        {
                Error message and Re-enter the id               // 9
        }
        for(i=0;i<machine_list_size;i++)                        // 10
        {
                Copy machine list in adjuster machines list  // 11
        }
}       // 12
```




**CFG For Add_Adjuster Function**

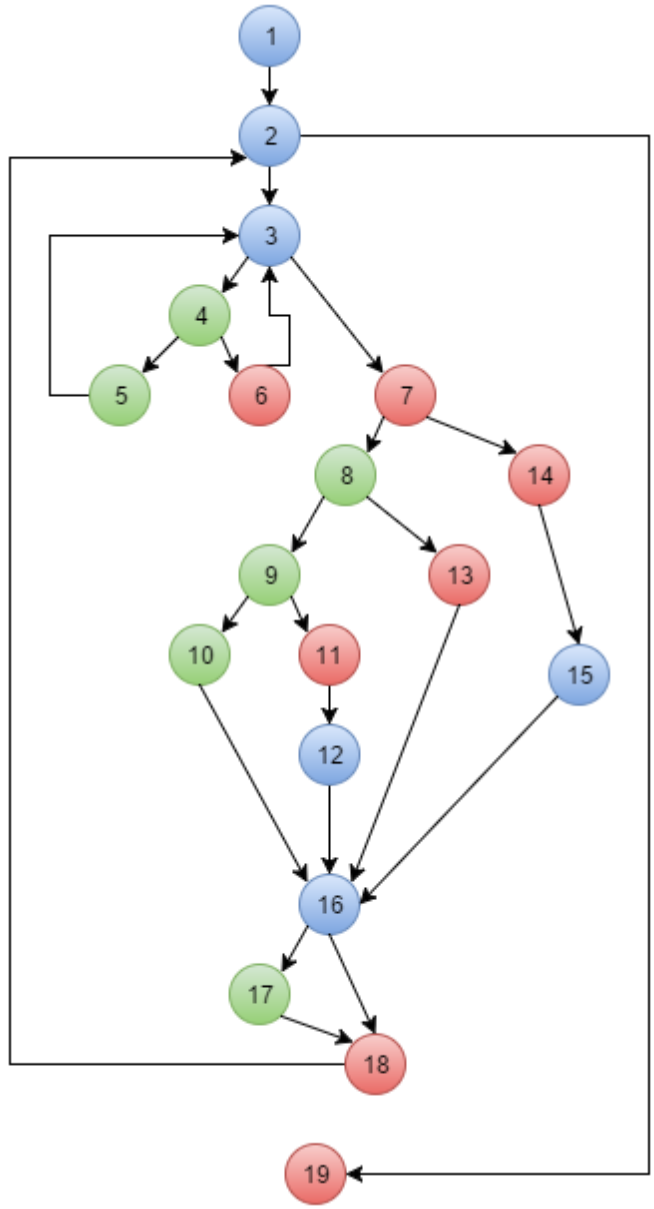
| | | |
|---|---|---|
| path-1 | 1-2-3-6-7-10-11-10-12 | quantity,id and machine list are copied to corresponding adjuster variables |
| path-2 | 1-2-4-5-6-7-10-11-10-12 | Quantity entered is out of bound// error message and then quantity is re-entered |
| path-3 | 1-2-3-6-8-9-10-11-10-12 | Id is not entered // error message and then id is re-entered |
| path-4 | 1-2-4-5-6-8-9-10-11-10-12 | Id and quantity error message// then both are re-entered. |

## Simulate Method:-

```
void simulate( machine_queue, adjuster_queue, simulation_period) //  1
{
        for(i=0;i<total_no_of_days;i++)//  2
        {
                for(machines in repairing state)//  3
                {
                        if(check whether machine repaired completely =yes)//  4
                        {
                                Update working status of machine//  5
                                Update working status of corresponding adjuster as
                                not working
                        }
                        Else ;          //  6
                }
                if( for each machine check machine failure=yes)//  7
                {
                        if(machine is not in repairing state=true)//  8
                        {
                                if(check whether adjuster available to repair that
                        machine=yes)//  9
                                {
                                        Assign adjuster to repair that machine
                                        Update adjuster status          //  10
                                        Status of machine as repairing
                                }
                                Else    //  11
```

```
                    {
                        Machine status-not repairing but failure//  12

                    }
                }
                Else;                   //  13
            }
            Else            //  14
            {
                Increase the count of machine running days by 1  //  15
            }
            if( for each  adjuster check whether he is working =yes)  //  16
            {
                Increase the count of adjuster working days by 1  //  17
            }
        }       //  18
}       //  19
```



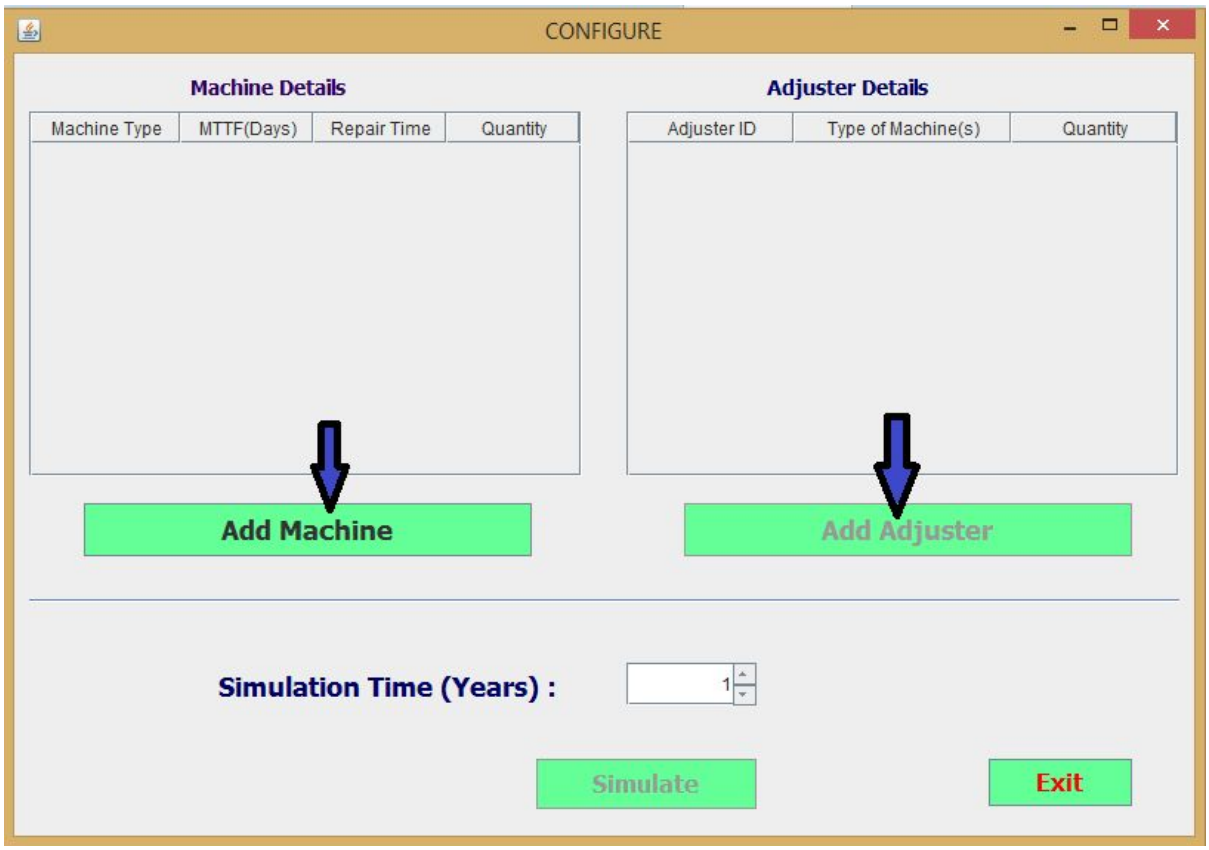

| | | |
|---|---|---|
| path-1 | 1-2-3-7-14-15-16-18-2-19 | This path gets followed when no machine gets failed in given simulation time |
| path-2 | 1-2-3-7-8-9-10-16-18-2-19 | This path gets followed when fault appear in any machine first machine |

Similarly other paths are also possible.

## USER INTERFACE TESTING:-

It has Included testing of GUI which is showing testing of each frame in GUI manually.

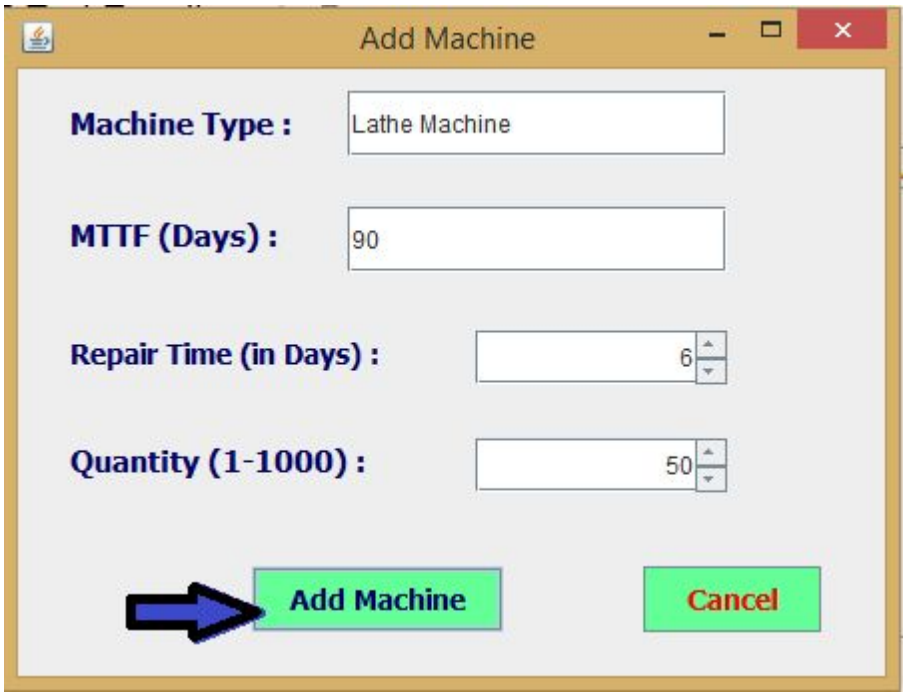

After Configure machines and adjusters are get added in the factory by add adjuster and add machines method as shown below.

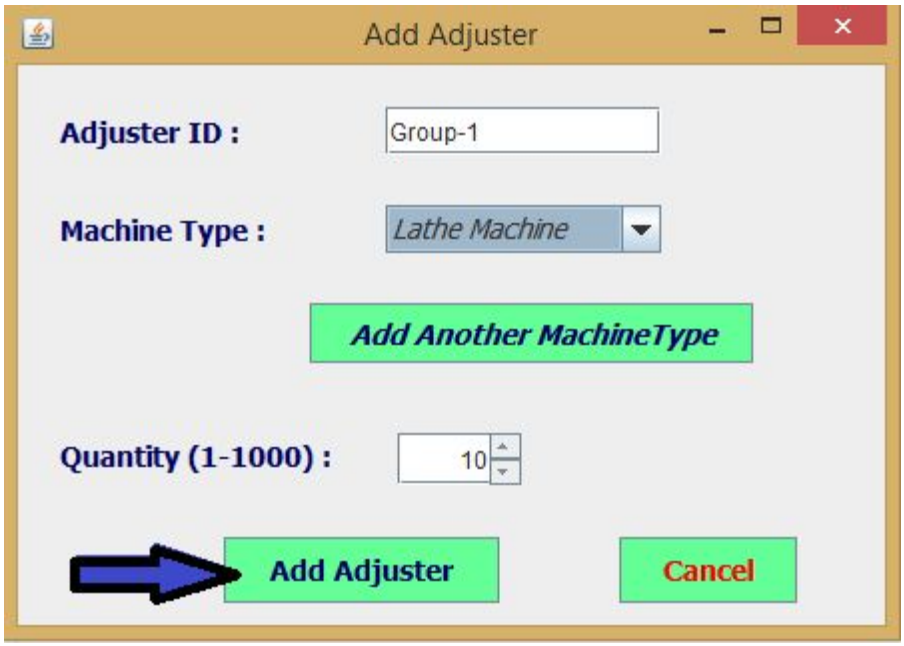



Add Machine - To add Machine

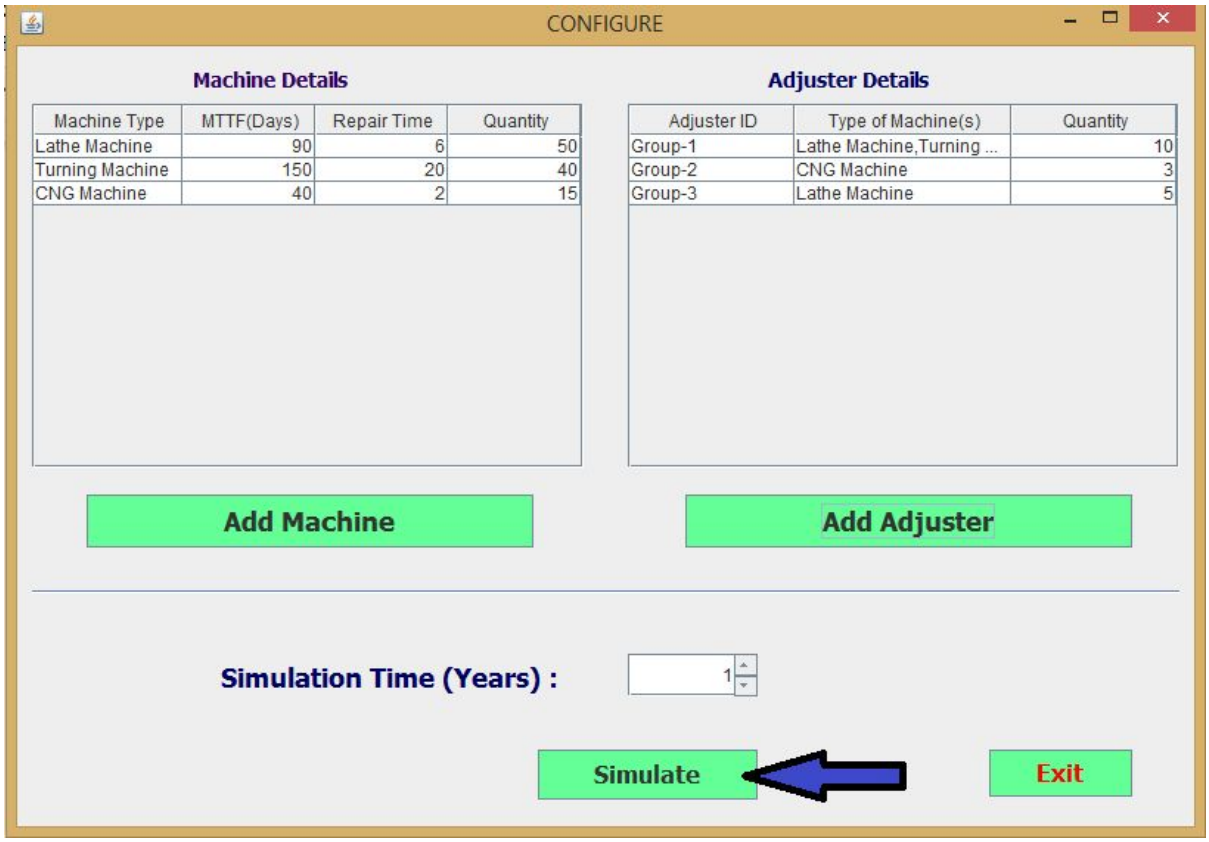
Add Adjuster- To add Adjuster

Details are entered for machine and for an adjuster.



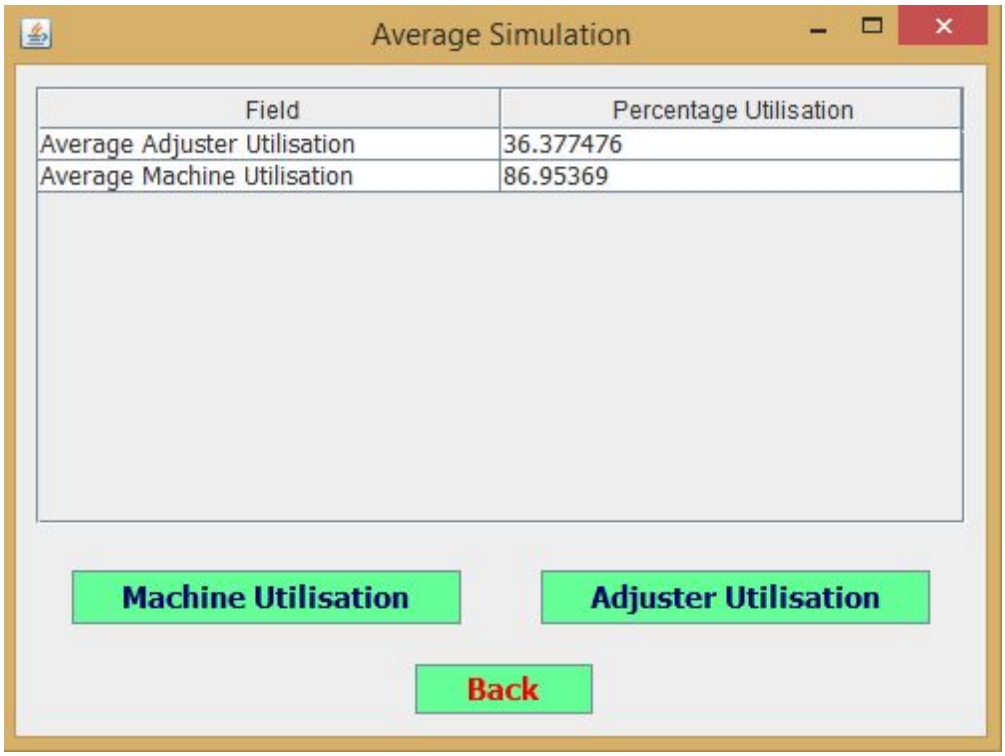




After clicking on Add Adjuster and on Add Machine button adjuster and machine get added in the respective queue.And queues are shown in GUI.

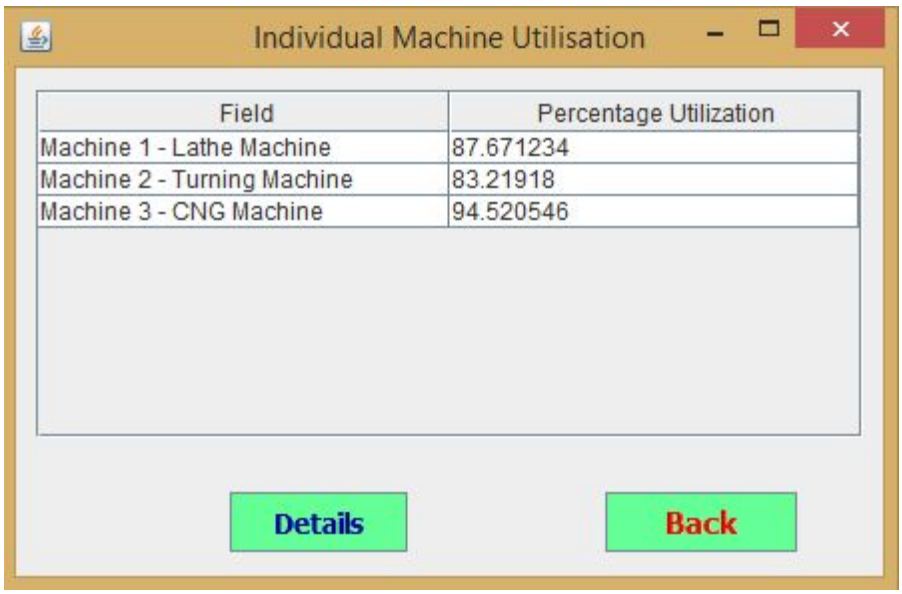


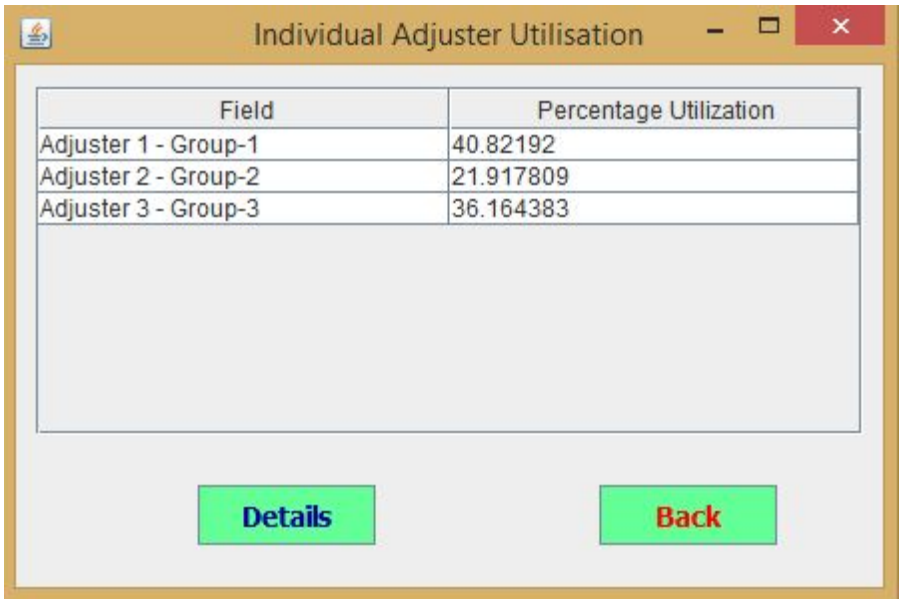

Simulation Time is Selected accordingly(in Years).

Average Adjuster Utilisation and Average Machine Utilisation percentage obtained after Clicking on Simulate Button.




To get utilisation of particular adjuster group and to get individual utilisation of particular machine Machine Utilisation And Adjuster Utilisation buttons  are used.






To get details about particular machine or particular adjuster group respective columns are selected in respective frames and details are obtained by using Details Button.

## Entry and Exit Criteria

This section describes the general criteria by which testing commences, temporarily stopped, resumed and completed within each testing phase. Different features/components may have slight variation of their criteria, in which case, those should be mentioned in the feature test plan. The testing phase also maps to the impact level definition when a defect is entered in the bug-tracking phase.

## Unit Testing

Unit Testing is done at the source or code level for language-specific programming errors such as bad syntax, logic errors, or to test particular functions or code modules. The unit test cases shall be designed to test the validity of the program correctness.

## Black Box Phase

Black box testing typically involves running through every possible input to verify that it results in the right outputs using the software as an end-user would. We will use Error Guessing and Boundary Value Analysis complexity metrics in order to quantifiably determine how many test cases needed to achieve maximum code coverage.

## Black Box Entry Criteria

The Black Box Entry Criteria will rely on the component specification, and user interface requirements. Things that must be done on entry to the Black Box stage:

- All functions like Add Machine, Add Adjuster, Get Simulation, Get Individual Machine and Adjuster, etc. must either be coded or stubs written.
- The type of Black Box testing Methods will be determined upon entry. We will use Error Guessing,and Boundary Value Analysis.
- Boundary Value Analysis included adding a high number (like 1000) of machines and adjusters simultaneously.

## Black Box Exit Criteria

The Black Box Exit Criteria listed below explains what needs to be completed in-order to exit Black Box phase. To exit the Black Box phase 100% success rate must be achieved. Things that must be done upon exiting the Black Box stage:

- The application showed no results in case of garbage string.
- If a Machine ID existing in the Table was entered at the time of adding a new product, an error message shown.
- All code bugs that are exposed were corrected whenever possible.

## White Box Phase

The White Box criteria apply for purposes of focusing on internal program structure, and discover all internal program errors. Defects were categorized and the quality of the software was assessed.

## White Box Entry Criteria

The White Box Entry Criteria relied on verifying that the major features work separately but not necessarily in combination. The design and human interface were stable. Things that were done on entry to the White Box stage:

- Unit tests were written for as many functions as possible.
- The type of White Box testing Methods that were used were determined upon entry. We used unit testing and test for memory leaks.
- Black Box Testing was in its late stages.

## White Box Exit Criteria

The Factory Service Simulation Software in the White Box stage generally had a stable feel to it. White Box testing continued until the Black Box or next milestone criteria were met. To exit the White Box phase 100% success rate was achieved. The following describes the state of the product upon exit from the White Box stage:

- All functions like Add Machine, Add Adjuster, Get Simulation, Get Individual Machine and Adjuster were implemented, operational and tested.
- All test cases were generated. The test cases were generated from the Control Flow diagrams of all functions.
- All code bugs that were exposed were corrected.

## System Test

The System Test criteria apply for purposes of categorizing defects and the assessing the quality level of the product. All elements of the Controller and Graphical User Interface were meshed together and tested as a whole. System test focuses on functions and performance, reliability, instillation, behavior during special conditions, and stress testing.