|  |  |
| --- | --- |
| **School of Computing & Digital Media** |  |
|  |  |

**Group 1 Individual Report:**

UK Building Maintenance Services

**Student Names:** Gheorghe Mitrea (17014657

**Course:** BEng (Hons) Software Engineering (Top-up)

**Lecturer:** Dr Tingai Wang

**January 2018**

Table of Contents

[1. Implementation 3](#_Toc505890306)

[1.1 AuhoriseEngineer class: 4](#_Toc505890307)

[1.2 DeRegisterEngineer class 5](#_Toc505890308)

[1.3 AddCatgory class 6](#_Toc505890309)

[4.4. ApproveJob class 8](#_Toc505890310)

[2. Testing 10](#_Toc505890311)

[2.1 Testing Design 10](#_Toc505890312)

[2.1.1 Test Design Techniques used to test the functionalities: 10](#_Toc505890313)

[2.2 Test Implementation 13](#_Toc505890314)

[**2.2.1** **Test User Requirements** 13](#_Toc505890315)

[**2.2.2** **Test Cases:** 14](#_Toc505890316)

[2.2.3 Build the Solution test 33](#_Toc505890317)

[2.2.4 Debugging test: 33](#_Toc505890318)

[2.2.5 Exploratory Test 34](#_Toc505890319)

[**2.2.6** **Black-box Testing** 35](#_Toc505890320)

[2.3 Test Summary Report 36](#_Toc505890321)

[3. User manual 39](#_Toc505890322)

[4. System Documentation 43](#_Toc505890323)

[5. Future Recommendations 45](#_Toc505890324)

[Conclusion 46](#_Toc505890325)

[References 47](#_Toc505890326)

**List of Figures**

[Figure 2 Code for AuthoriseEngineer class 5](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890270)

[Figure 2 Authorize engineer 5](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890271)

[Figure 3 De-register an authorised engineer 6](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890272)

[Figure 4 Code of DeRegisterEngineer class 6](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890273)

[Figure 5 AddCategory functionality 7](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890274)

[Figure 6 Code for AddCategory class 8](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890275)

[Figure 7 Functionality of ApproveJob form 9](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890276)

[Figure 8 Code of ApproveJob class 10](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890277)

[Figure 9 Naming violation rules 11](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890278)

[Figure 10 Engineer successfully authorized by admin 18](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890279)

[Figure 11 Successful de-registration by admin 19](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890280)

[Figure 12 Validate the De-authorize function by trying to log in back with the engineer's credentials 20](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890281)

[Figure 13 A created job by a customer is approved by the admin and assign it to an engineer 21](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890282)

[Figure 14 Category added successfully by admin and validated into the Category list 22](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890283)

[Figure 15 Cancel a job by a customer 23](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890284)

[Figure 16 Job was created successfully by customer1 24](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890285)

[Figure 17 The new created job by customer 1 was successfully added to table tbl\_job of the ukservices database 25](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890286)

[Figure 18 Testing the functionality of adding a feedback to a completed job was validated successfully 25](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890287)

[Figure 19 Validating the functionality of Login 26](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890288)

[Figure 20 The registered engineer but not authorized yet by the admin 27](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890289)

[Figure 21 Unsuccessful login of a registered engineer but not authorised yet by admin 27](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890290)

[Figure 22 Validate the registration of a new customer 28](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890291)

[Figure 23 Registered customer Monaco were added to the database to table tbl\_customer 29](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890292)

[Figure 24 Validate the functionality of selecting a job by an engineer which wasn't assigned by customer 29](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890293)

[Figure 25 Validating the functionality of updating a job status 30](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890294)

[Figure 26 Validation of viewing the status of a job by a customer 31](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890295)

[Figure 27 Validating the Weekly Report functionality by admin 32](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890296)

[Figure 28 Weekly report to be saved and printed by admin 33](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890297)

[Figure 29 Rebuild the solution to find any errors and warnings 34](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890298)

[Figure 30 Debugging test showing no errors and few messages of naming violation rule 34](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890299)

[Figure 31 Warning about boundary for branches exceeded 35](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890300)

[Figure 32 New Boundary fixed for max limit of 20000 branches 35](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890301)

[Figure 33 New Boundary fixed for max limit of 20000 branches 36](file:///E:\Top%20Up%20year\Course%202017_2018\Advanced%20Software%20Engineering\17014657-part%201.docx#_Toc505890302)

**List of Tables**

[Table 1 Test Cases 15](#_Toc505890303)

[Table 2 Black-box testing 36](#_Toc505890304)

[Table 3: System Documentation 44](#_Toc505890305)

# Implementation

The customer service system has been designed using C# programming language. System implementation describes about the physical system design of a proposed system. It describes about how the system should be built. It is used to determine the operations performed by a system. It describes the quality assurance of the system (Uky, 2017).

* Conceptual: it provides graphical represents of a system. It determines the entities, attributes and relationship between each entity. Conceptual design is the basic layout of the system that is designed to collect the user requirements for the designing of a system.
* Logical: this is used to determine the flow of information in the system. It provides the general idea of what the system will work for the users.
* Physical: it determines how the system will perform business operations.

The next step is to implement the functionalities of authorise an engineer by implementing AuthoriseEnginner class; de-authorise an engineer by implementing DeRegisterEngineer class; add a speciality to a registered engineer by implementing AddCategory class

.

## AuhoriseEngineer class:

The scope of this class is to implement the functionality of authorising a registered engineer to take jobs from customers and being approved by the Admin of the system. The form of authorising an engineer include the details of EngineerID, Name of the engineer, Address, Mobile No, Email ID and Specialization (work category such as painting, roofing, plumbing etc.):



Figure 2 Code for AuthoriseEngineer class

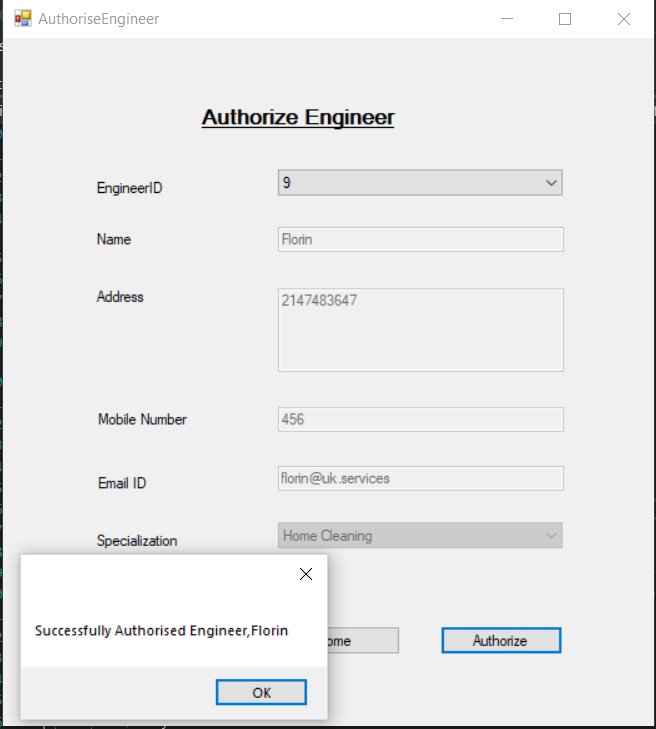


Figure 2 Authorize engineer

## DeRegisterEngineer class

The scope of this class is to de-register an authorised engineer from the engineers list. This functionality contains all the details of an authorised engineer by choosing the ID of the engineer and all the details will be displayed to the admin:

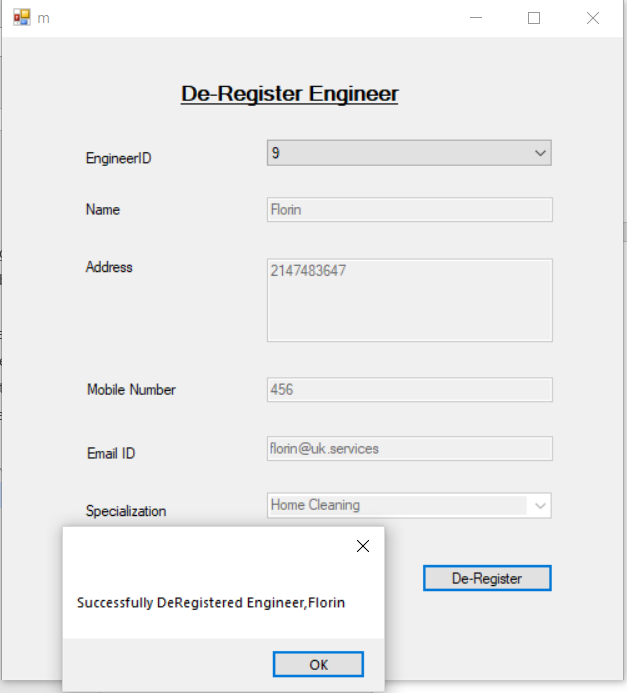


Figure 3 De-register an authorised engineer

A piece of code for this form is showing in the next figure:

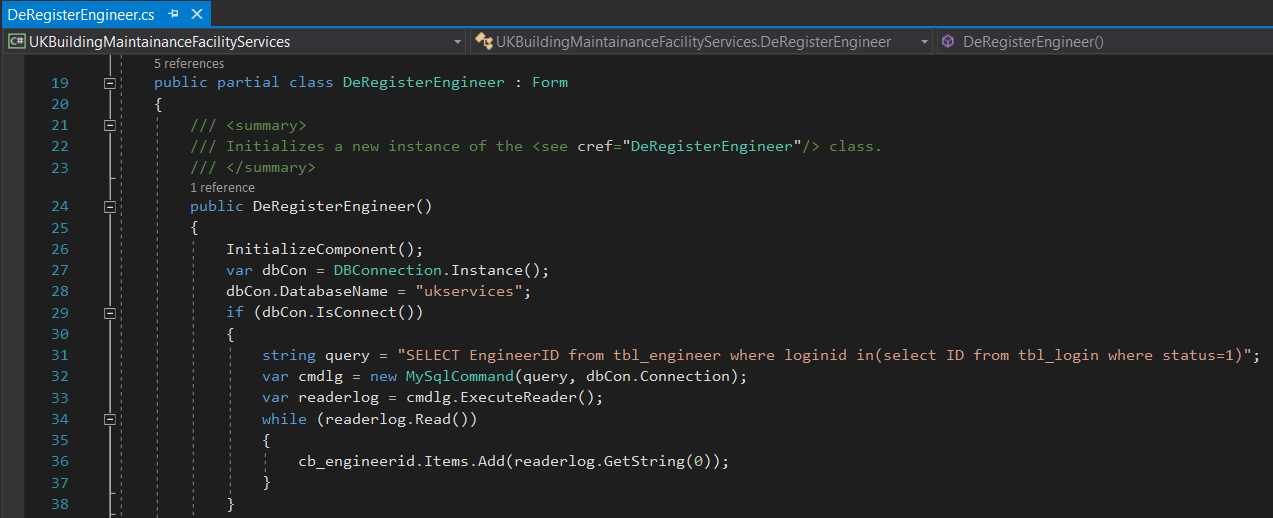


Figure 4 Code of DeRegisterEngineer class

## AddCatgory class

The scope of this class is to add a job category to the list of engineers’ specialisations, to be easier for customers and for admin to select an engineer for an adequate job.

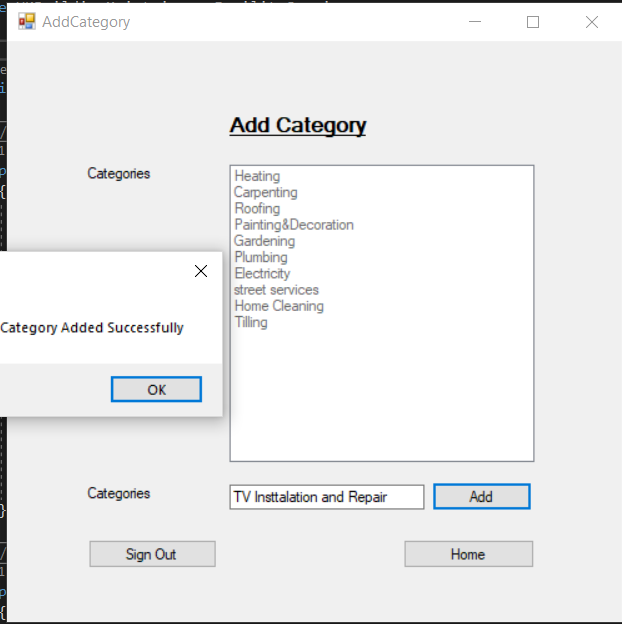


Figure 5 AddCategory functionality

A piece of code for AddCategory class is shown below:

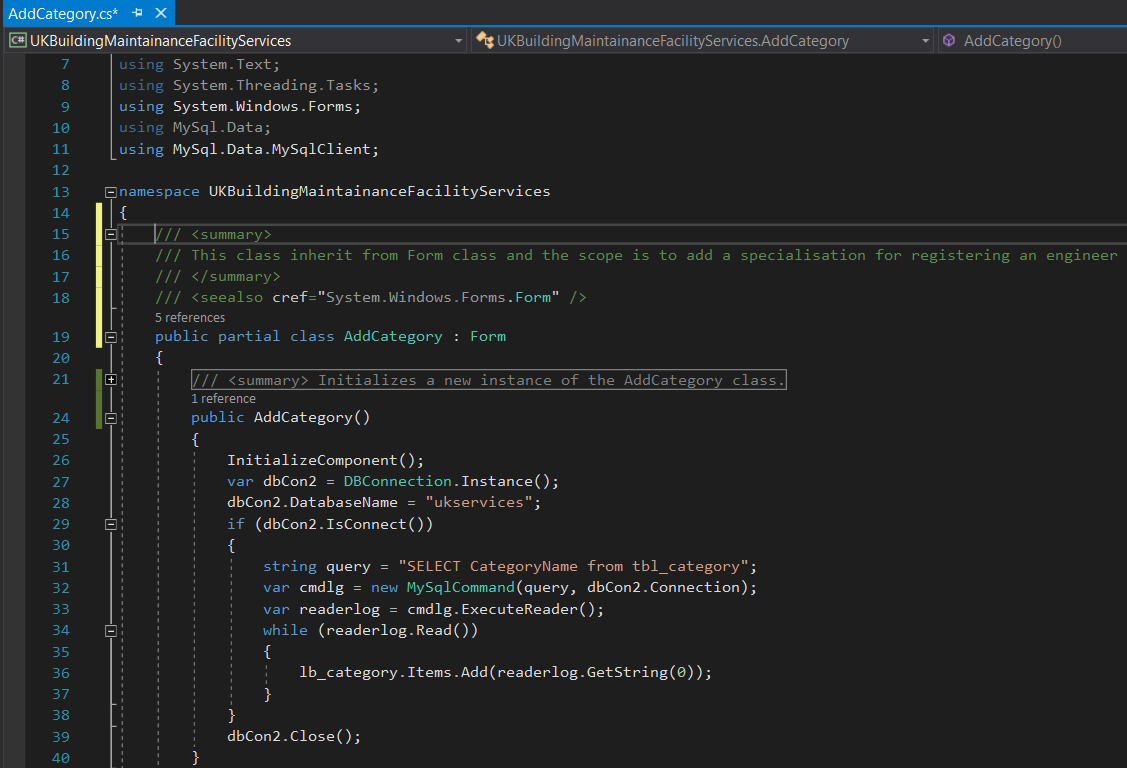


Figure 6 Code for AddCategory class

## ApproveJob class

The scope of this class is to implement the functionality of approving the job ID by the admin with all its details supplied by the customer, where it can be find also the selected engineer for that job.

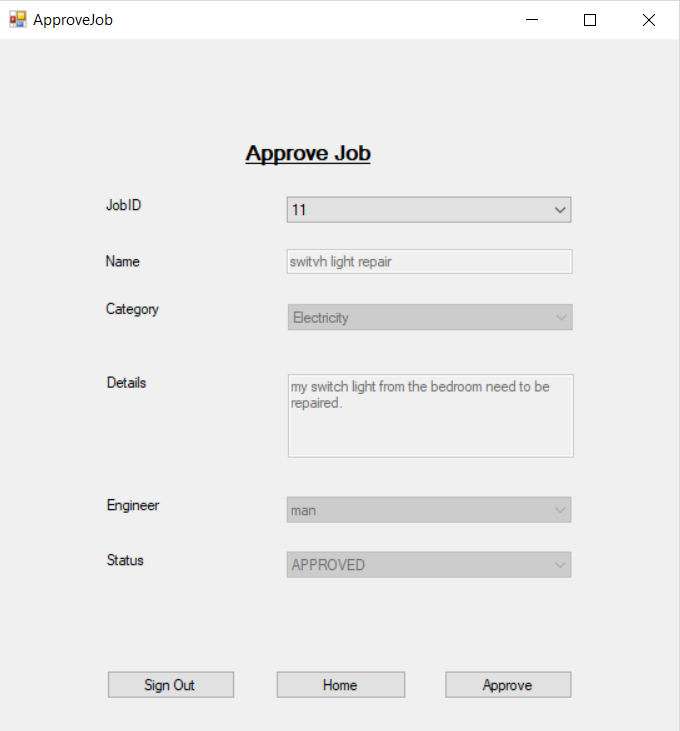


Figure 7 Functionality of ApproveJob form

A piece of used code to implement this class is shown in the below figure:

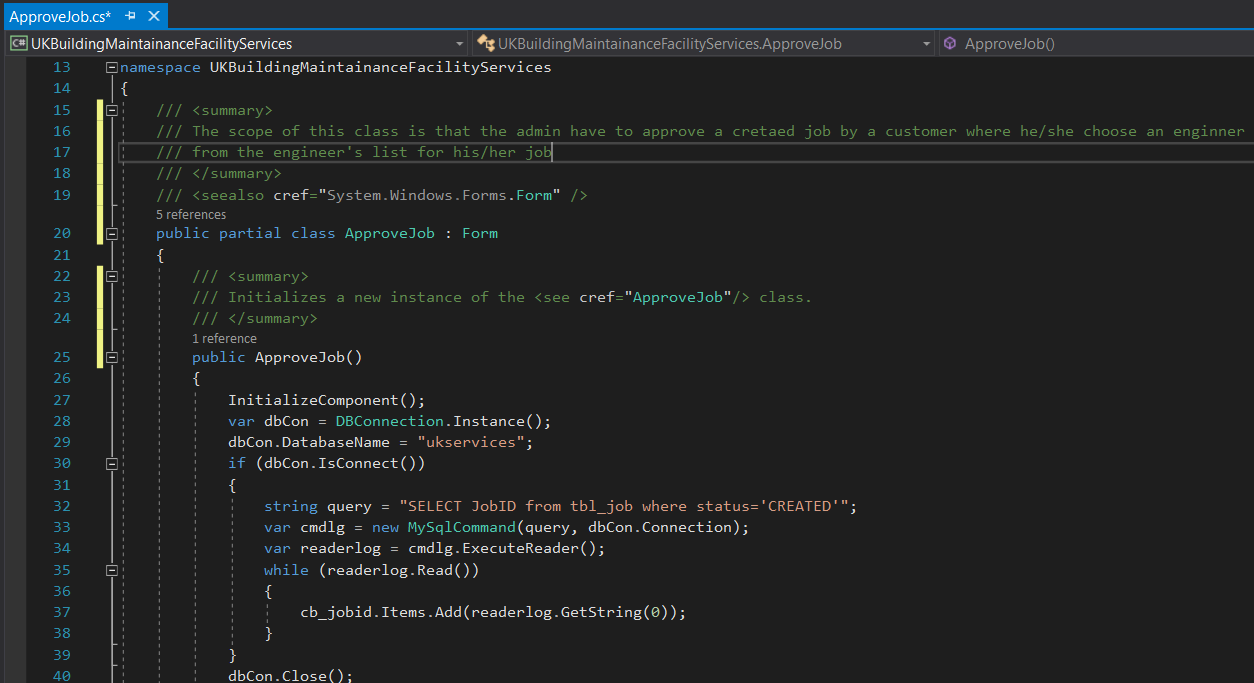


Figure 8 Code of ApproveJob class

# Testing

The testing processes consists in testing design, implementing the tests and Test Report for all the requirements.

## Testing Design

### Test Design Techniques used to test the functionalities:

Were reviewed the requirements and specifications of the test plan. The objectives of this phase were to define the test design specification, tests cases specifications and the test procedure. Were created the test cases and test data.

During the Testing Design phase were used both tests such as Static Test and Code Review for all implemented functionalities without executing the code, were done manually.

At this stage the developer made a verification which implies the Static Testing of all implemented classes where he found some naming violation rule of local variables in the source code such as ***btn\_signout\_Click*** from ApproveJob class, where words must begin with upper characters:



Figure 9 Naming violation rules

The system is designed adopting a test-driven approach; each unit test can be seen as a design component defining classes, methods, and noticeable behaviour.

#### White-box testing method:

The test case as shown in tabular form is done to check whether the developed system works properly or nit before deploying it into the client’s environment. By the test results, it is clear that the customer service system for the UK building Maintenance Company covers all the required functional and non-functional requirements as stated in the Group Report:

1. Login/register- new engineers have to register themselves into the application to access the services. The existing users have to enter their valid email address and password to continue accessing with the services.
2. De-register an engineer by the admin - This operation is required to eliminate the details of an engineer who no longer works with the company or was sucked. Or it can be used to eliminate any wrong entry in the system.
3. Create job- There has to be an option ‘new’ to create a new entry in the system for a specified engineer. Second option should be to update the details of a particular engineer.
4. Customer Feedback- customers have to login first to provide their valuable feedback for the services accepted from the company.
5. Add a specialization to an engineer when he is registering
6. Approve a job by the admin when it is created by a customer
7. Authorise or de-authorise an engineer by the admin when the engineer registered
8. Cancel an on progress or created job by a customer or by an engineer
9. Generate weekly reports of jobs with all the afferent details of it
10. Select job by an engineer when the job is not assigned to any engineer by the customer
11. Update job status by an engineer
12. View the job status by a customer

The screenshots for each test case is shown below in the Test Implementation

#### Black-box testing:

To test the developed system, functionality test method is used. It is a type of black box testing method that is used to assure the quality of a developed system. It is used to verify the different functions that are performed by the system. In functional testing, following functions are tested (Guru99, 2017):

* Mainline functions
* Usability of system which includes proper navigation
* Accessibility to ensure that each user can access the system without any technical issues.
* Error to ensure that the system is reliable to use

## Test Implementation

Test implementation is done by using dynamic testing to validate all the user requirements, functional and non-functional requirements stated above and also in the Group Report.

### **Test User Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| Role of User | Type of User | Functionality needed | Test Functionality |
| Company Administrator | Creator | Register/deregister a new engineer  Create login page for different users like customer and engineer | - The system is able to Register/deregister a new engineer  - The system is able to create login page for different users like customer and engineer |
| Manager | Contributor | Create a job  Cancel a job  View progress reports | The system is able to create a job, cancel a job or view the progress reports |
| Engineer | Contributor | Mark completed jobs |  |
| Customer | Consumer | Request for service  Take service  Leave feedback | A customer can make requests for services, can take a service and leave a feedback for a job done by an engineer |

### **Test Cases:**

Table 1 Test Cases

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Expected Result | Actual Result | Pass or Fails |
| Add category | Customers are able to add category for job | Customers are able to add category for job as expected | Pass |
| Approve job | Users click on approve button to approve a job using ‘approve’ button | Buttons are functioning properly as the job is approved when users click in ‘approve’ option as expected | Pass |
| Authorization and de-authorization of engineers by admin | After successful registration of an engineer, he will only be able to login to system after getting authorized by admin | Admin can authorize/de-authorize engineers as expected | Pass |
| Cancel a Job | Cancel a job with a status of Created, Approved or On Progress job status | Customers can successfully cancel a created, approved or ongoing job | Pass |
| Customer Feedback | Customers can provide feedback for the running job | Customers can provide feedback for the running job as expected | Pass |
| De-registration of an engineer | Admin can de-register an engineer | Admin can de-register an engineer as expected | Pass |
| Successful Login | Admin can successfully login by using valid username and password | Admin can successfully login by using valid username and password as expected | Pass |
| Customer Registration | Successful registration of customer with complete login details | Successful registration of customer with complete login details as expected | Pass |
| Engineer Registration | Successful registration of engineer with complete login details | As expected, engineers are successfully registered in the system | Pass |
| Select Job | When a job is not assigned to an engineer by a customer then any engineer can take that unassigned job by selecting the ID of the job with all its details | Any engineer can successfully select an unassigned job of a customer by selecting the Job ID with all its details | Pass |
| Job status Update | Engineers can update the status of a job which is in progress or yet to begin | Engineers can successfully update the status of a job which is in progress or yet to begin as expected | Pass |
| View Job status | A customer can view the status of his/her created job | Any customer can successfully view the status of their created job | Pass |
| Create job option | Customers can choose different options and create a job | Job is successfully created as expected by the customer | Pass |

#### **Execute Test Cases**

1. **Authorise an engineer** to take jobs from customers or to be selected by the customers:

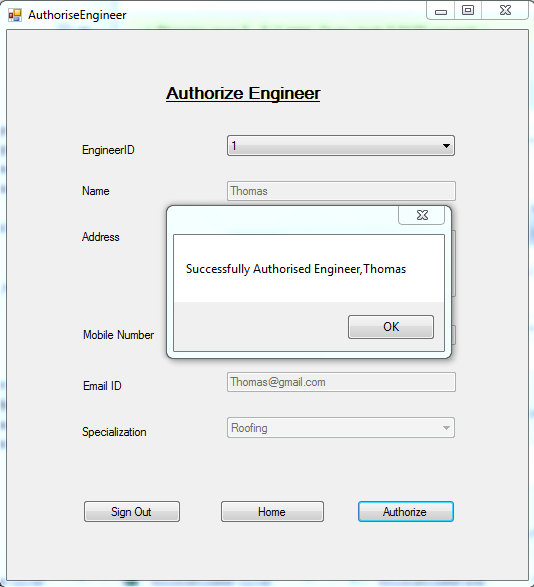
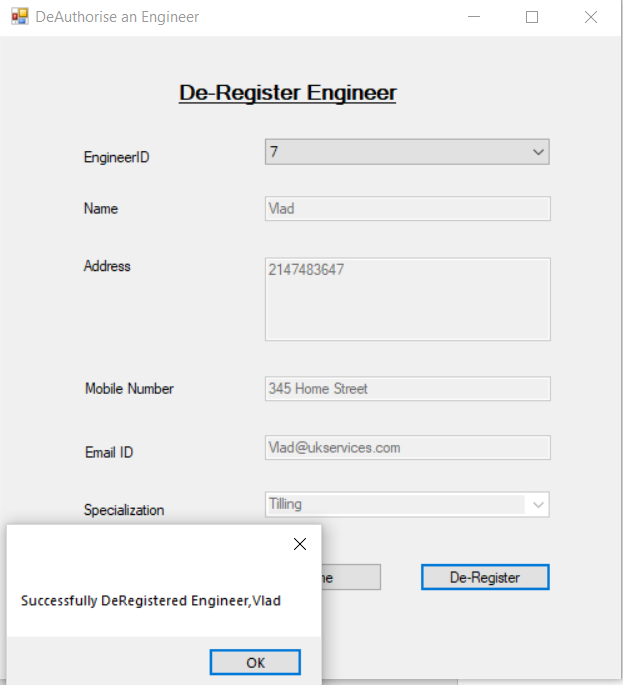


Figure 10 Engineer successfully authorized by admin

As it can be seen from the above figure the expected result of authorizing an engineer was validated by a message box that engineer with ID 1 was authorized by the admin to take jobs from customers.

1. **De-register an engineer** – remove details of a specified engineer (with login password)

Figure 11 Successful de-registration by admin



As it can be seen from the above figure a validation of De-register an authorised engineer was successfully tested by selecting the engineer to be removed from the engineers’ list.

Engineer Vlad, who was De-authorised by the admin, tried to log in into the application using his username ‘Vlad’ and password ‘123’ but a box message is displayed “Invalid credentials” validating the function of De-registration an engineer by the admin:

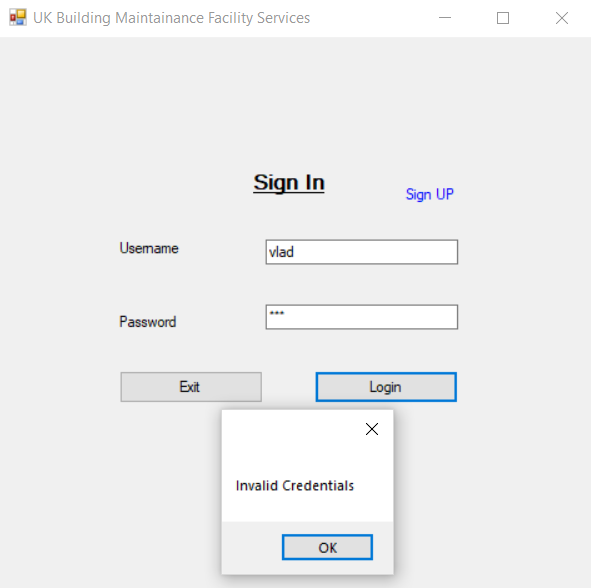


Figure 12 Validate the De-authorize function by trying to log in back with the engineer's credentials

1. **Approve jobs** by the admin when it was created by customers to be given to an engineer

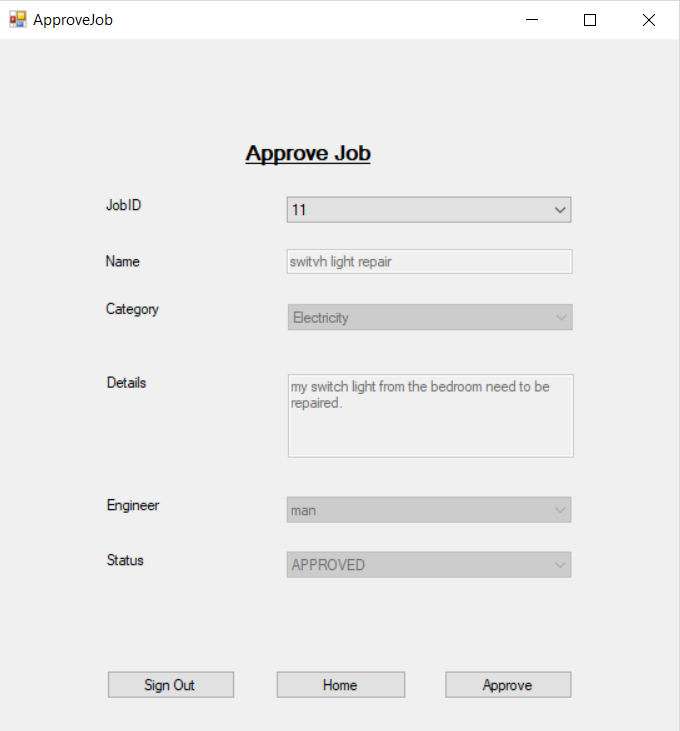


Figure 13 A created job by a customer is approved by the admin and assign it to an engineer

As it can be seen from the above image a validation of approved job by the admin was successfully.

1. **Add a category** of engineer’s specialisation from where a customer or the admin can see his skill.

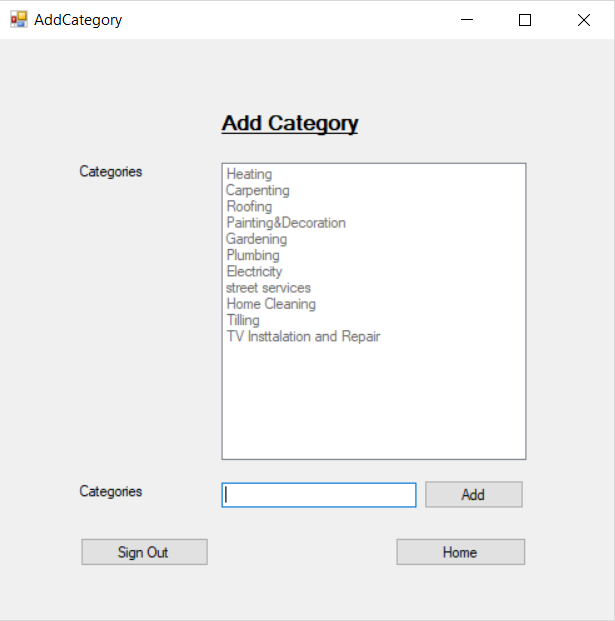
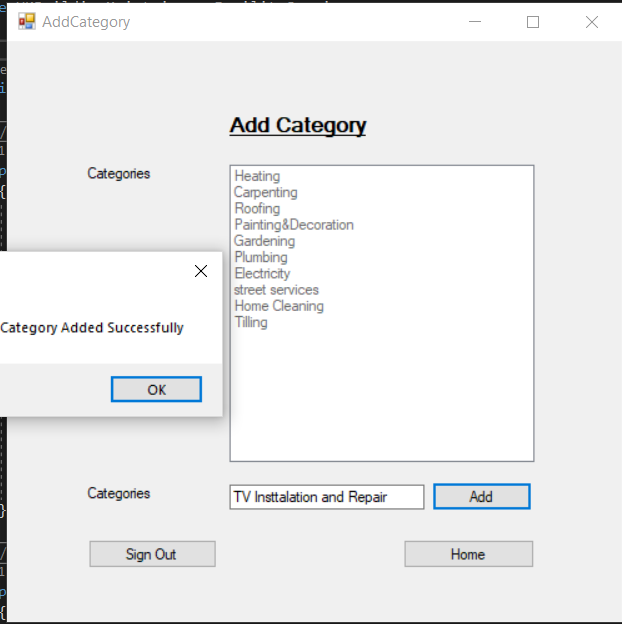


Figure 14 Category added successfully by admin and validated into the Category list

Testing the functionality of adding a new specialization to the categories’ list was successfully validated by input a new specialisation “TV Installation and Repair” with the expected outcome result that “Category Added Successfully” to the list as it can be seen from Figures 23 and 24:

1. **Cancel a job –** This test consists in validating the functionality of cancelling a job by the customer who created it when is on progress or is just created:

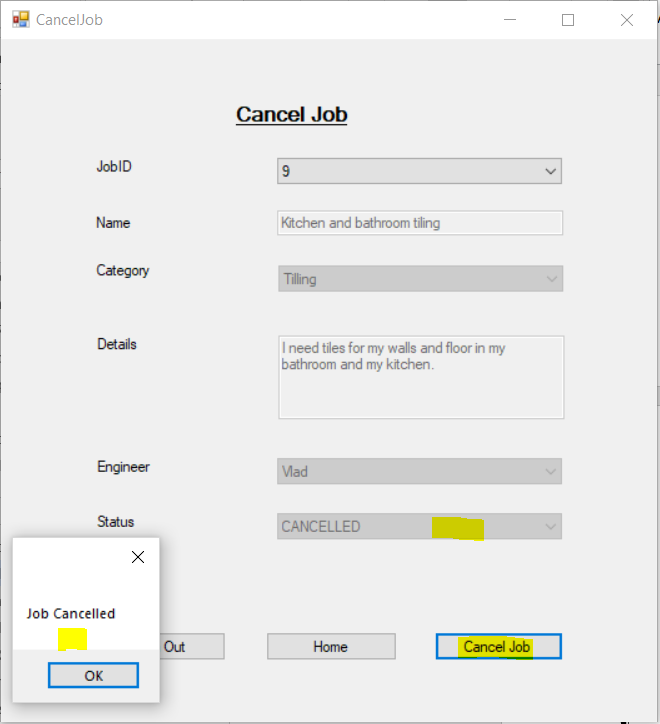


Figure 15 Cancel a job by a customer

Testing the functionality of cancelling a job of a customer was successfully validated by choosing the right job ID which the customer wanted to cancel it by pressing Cancel Job button from CancelJob form followed by text message which confirmed the expected output and the status of the job.

1. **Test of Create Job –** This test consists of validating the functionality of creating a job by a customer:

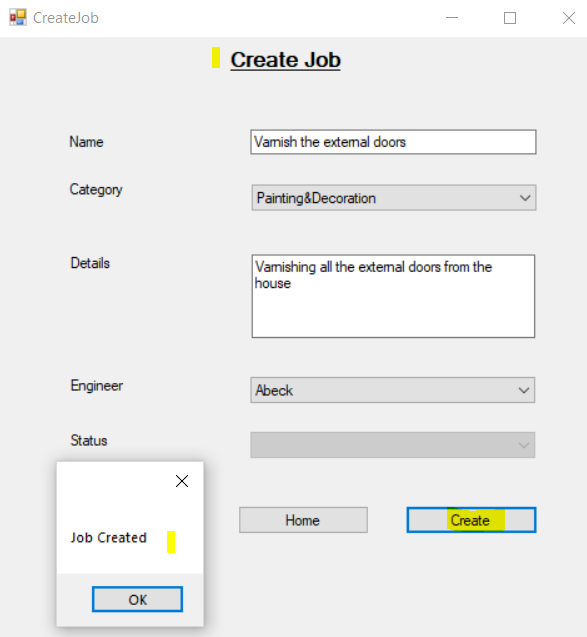


Figure 16 Job was created successfully by customer1

In the above figure it can be seen that the validation of creating a job by a customer named customer1 was successfully. This was done by introducing the details of the new customer’s job, such as name of the job, category of work area, details of the job to be done, selecting an engineer or not for that job. After all these details were introduced, the customer must press Create button to successfully create the new job. Was successfully tested and validated this functionality.

Also, the functionality was verified by checking if all the details of the new job created by customer1 were introduced correctly into ukservices database for table tbl\_job where all the created jobs should appear.

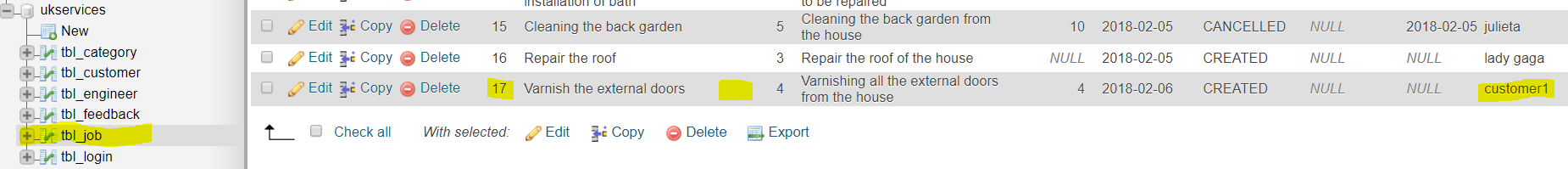


Figure 17 The new created job by customer 1 was successfully added to table tbl\_job of the ukservices database

1. **Customer Feedback functionality –** This test consists in validating the functionality of creating and sending a feedback regarding the job done by an engineer. The Feedback form include the details of job ID, the name of the engineer, the feedback’s description and rank the engineer’s job done for the customer. After all the details were introduced was pressed Send Feedback button

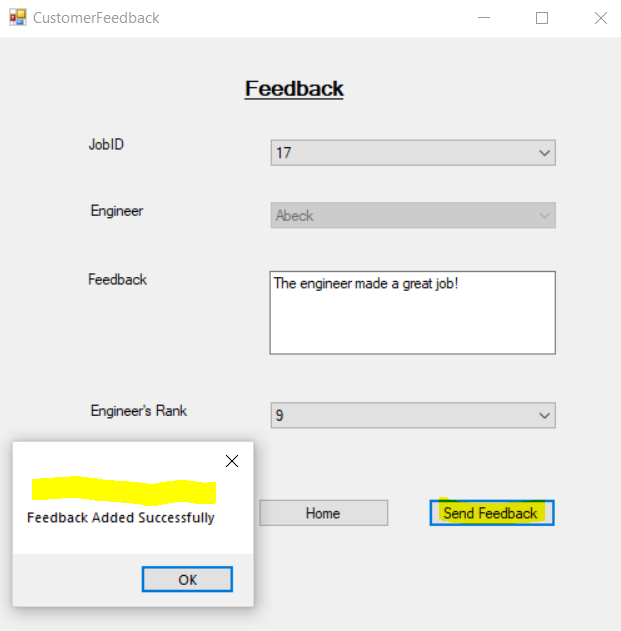


Figure 18 Testing the functionality of adding a feedback to a completed job was validated successfully

To validate a feedback from a customer related to the job ID done by an engineer, was necessary to fill all the requirements of Feedback form and then clicked Send Feedback button to see if the functionality is coming with the expected outcome as the message “Feedback Added Successfully”. The test passed as is shown in Figure 18.

1. **Testing the Login or Log out functionality** – this test consists of validating the functionality of Login or Log out from a user account:

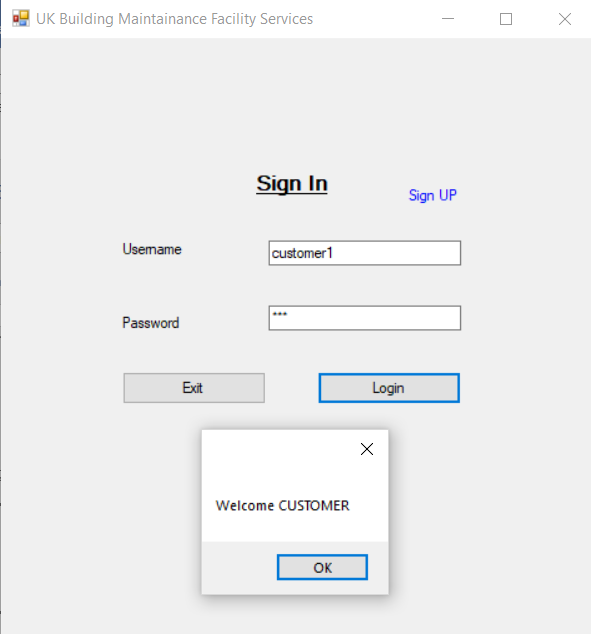


Figure 19 Validating the functionality of Login

As it can be seen from the above screen shots of Figure 19 the validation of Login functionality was done by input a user credentials such as the username and the password of it. If the inputs were correctly introduced then he should be pop-up into Customer Home window (as in the above images), Engineer Home window or Admin Home window.

If the user input wrong credentials, then the application should display a message of invalid login credentials of user such as in the below figure. Same message will be displayed when a registered engineer wants to login but is not authorised by the admin:

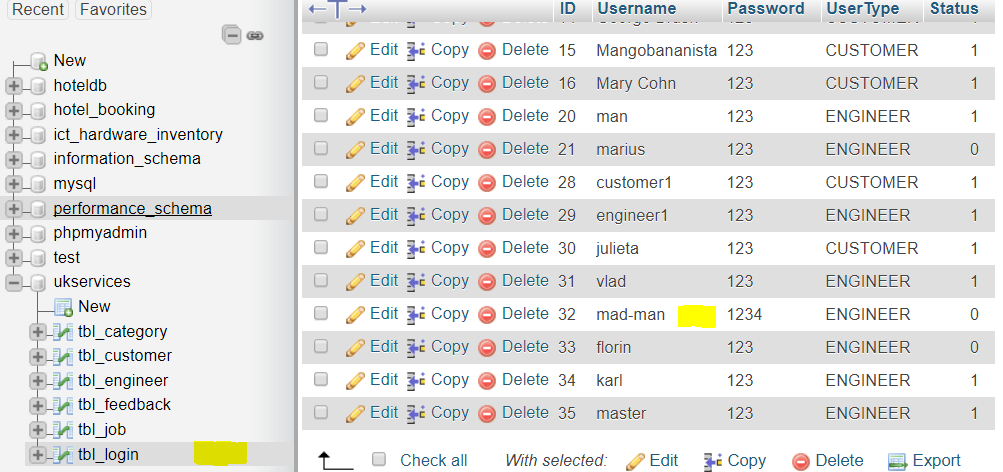


Figure 20 The registered engineer but not authorized yet by the admin

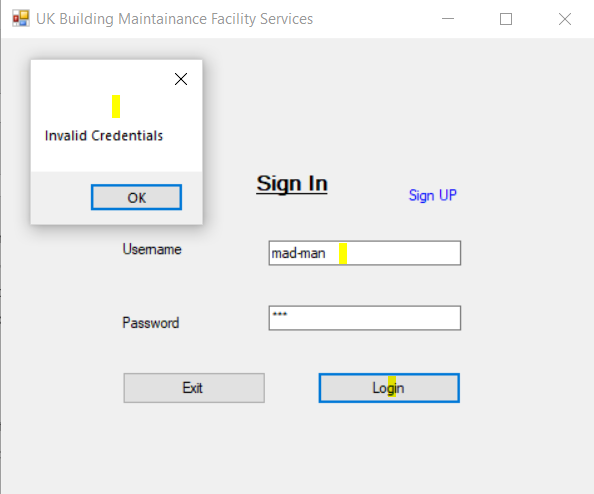


Figure 21 Unsuccessful login of a registered engineer but not authorised yet by admin

1. **Testing the Registration functionality -**  This test consists in validating the functionality of registering or sign up a new user (customer or engineer).

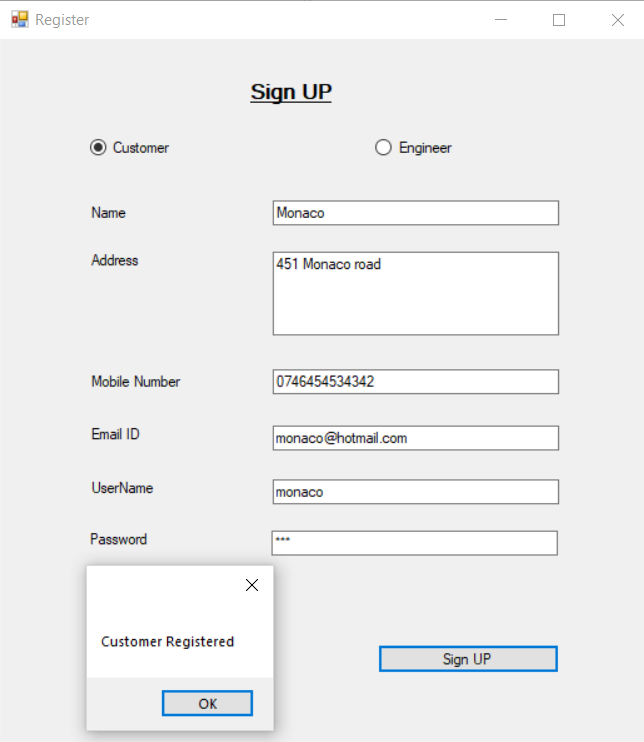


Figure 22 Validate the registration of a new customer

After fill all the details needed to register as a new customer the validation of this functionality displayed the expected outcome, a message “Customer Registered” showed.

The details of the new registered customer were added to the database for table tbl\_customer, the test passes with success:

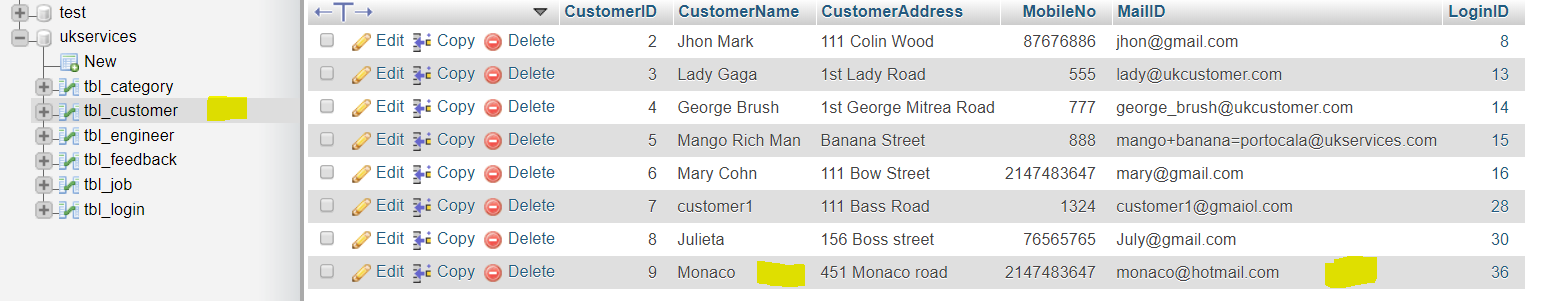


Figure 23 Registered customer Monaco were added to the database to table tbl\_customer

1. **Select Job testing functionality:** This test needs to validate the functionality of selecting a job by an engineer where the customer didn’t assign any engineer to it for that category work. The job needs first to be accepted before an engineer select it. The test consists in selecting the job ID with all its details from the Select Job window of the engineer account. After this by pressing Select button it should come with a message that the job was selected by engineer. The below figure confirms the success of validation for this functionality:

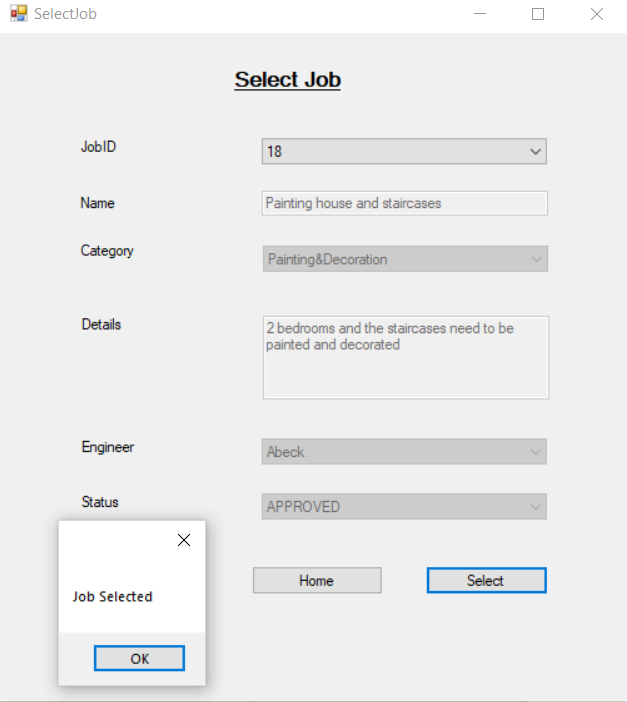


Figure 24 Validate the functionality of selecting a job by an engineer which wasn't assigned by customer

1. **Update Job Status testing functionality:** This test consists in validating the functionality of updating the status of a job which was assigned to an engineer by the customer. The below figure is showing a successfully validation for this functionality by selecting the required Job ID with all its details. After this the engineer must ensure that he is updating the status of the job depending on its status at that moment, which can be just created and needs to start, on going meaning that the job is under work by engineer, completed or even cancelled by the customer. For example, if the job is on going status and the engineer just finished the job then he must update the status of the job as Completed by pressing Update Status as is showing in the next figure:

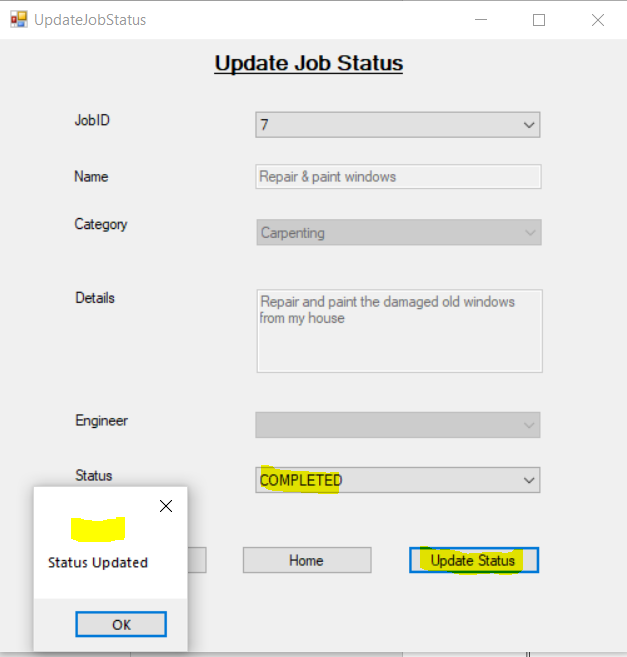
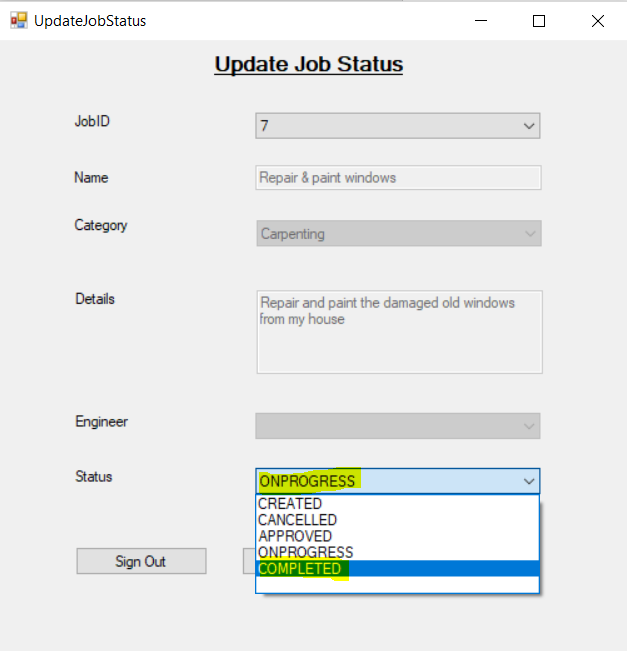


Figure 25 Validating the functionality of updating a job status

1. **View Job status testing functionality –** This test consists in validating the functionality of viewing the status of a job by a customer, where the engineer assigned to the job set a status for the requested job. The validation can be done by selecting the requested job ID with all its details and the current status for that job, in View Job Status window from the customer account as in the below figure:

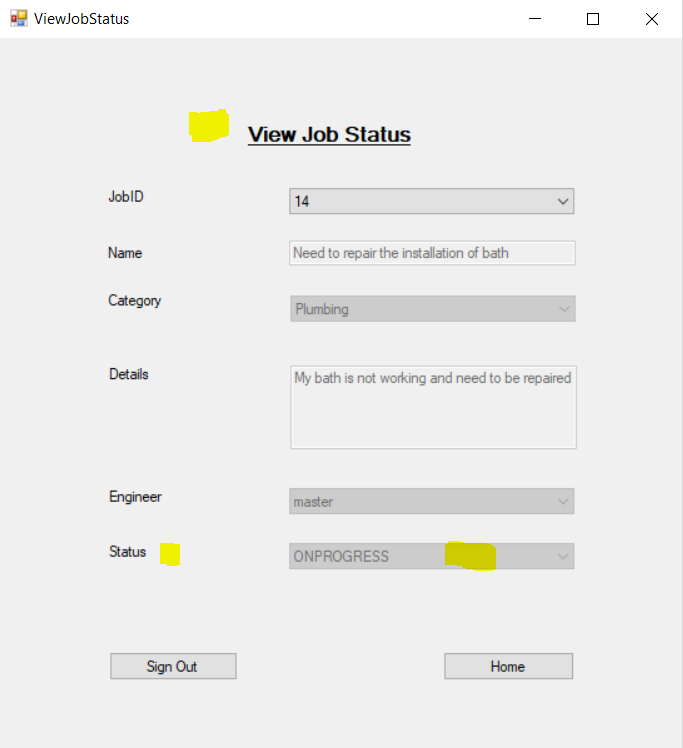


Figure 26 Validation of viewing the status of a job by a customer

1. **Weekly Report testing functionality –** This test consists in validating the functionality of generating a weekly report for all kind of jobs with all their details. The validation can be done by selecting a time period for which the admin can generate the report and where he must always input the start and the finish date for each job. After this was tested the functionality of saving the report in a PDF format and then the possibility to print it. As the below figure is validating all the expected result for this functionality then this test passed successfully:

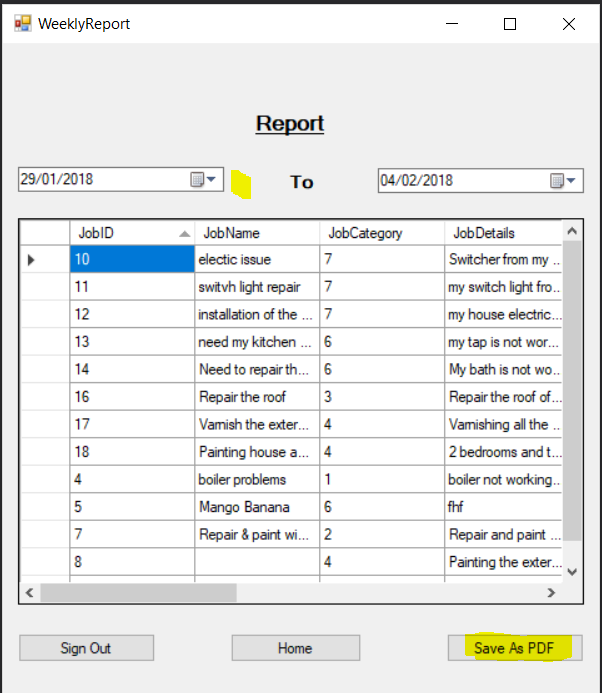


Figure 27 Validating the Weekly Report functionality by admin

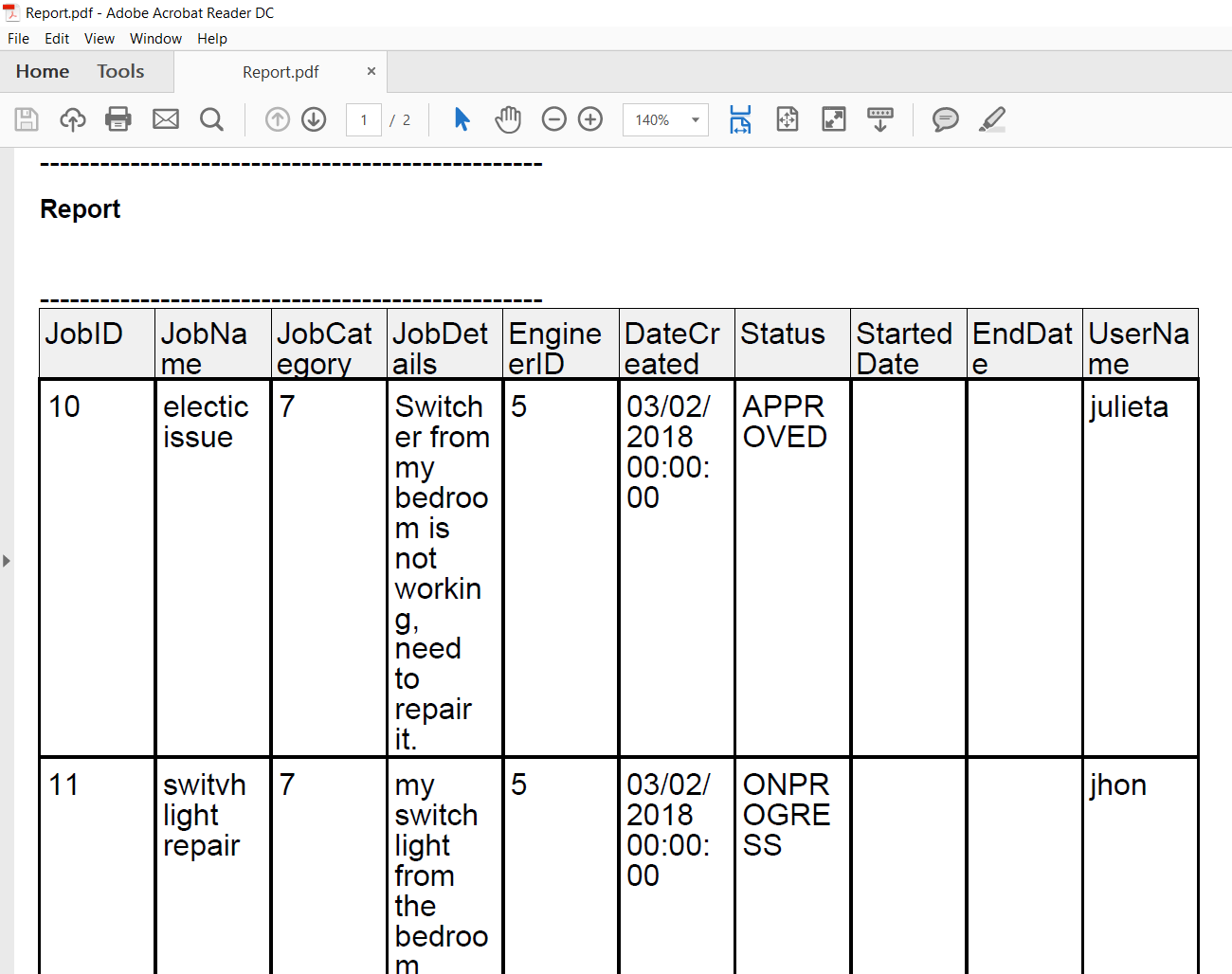


Figure 28 Weekly report to be saved and printed by admin

### Build the Solution test

The below figure shows the execution of Rebuild Solution which comes with no errors and two warnings related to add an Assembly Version to the solution and the other one related to the fact the an object can be disposed more than once in a method.

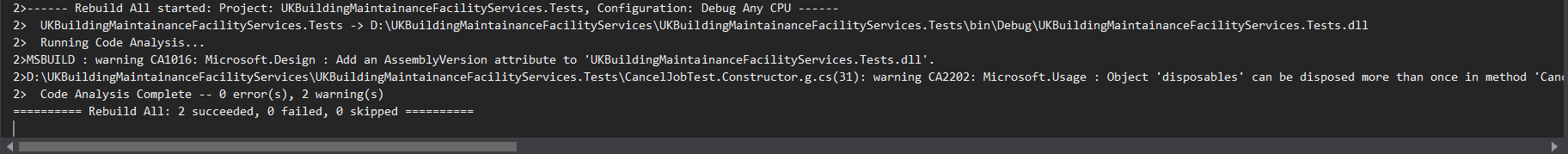


Figure 29 Rebuild the solution to find any errors and warnings

### Debugging test:

Debugging test is looking for an automation testing of finding and fix the bugs of the implemented system. The below figure shows that the application doesn’t have any errors after debugging less a few messages showing the violation of naming rule where the words must start with a upper case character as is showed in the below image:

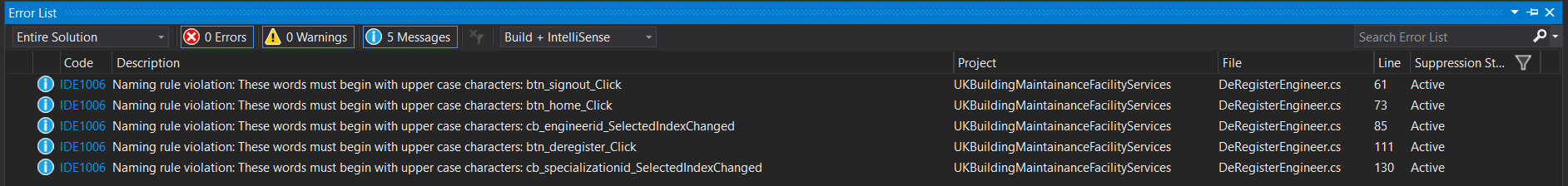


Figure 30 Debugging test showing no errors and few messages of naming violation rule

### Exploratory Test

The below figure shows the Boundary Test where the max value limit was 10000 branches and was fixed

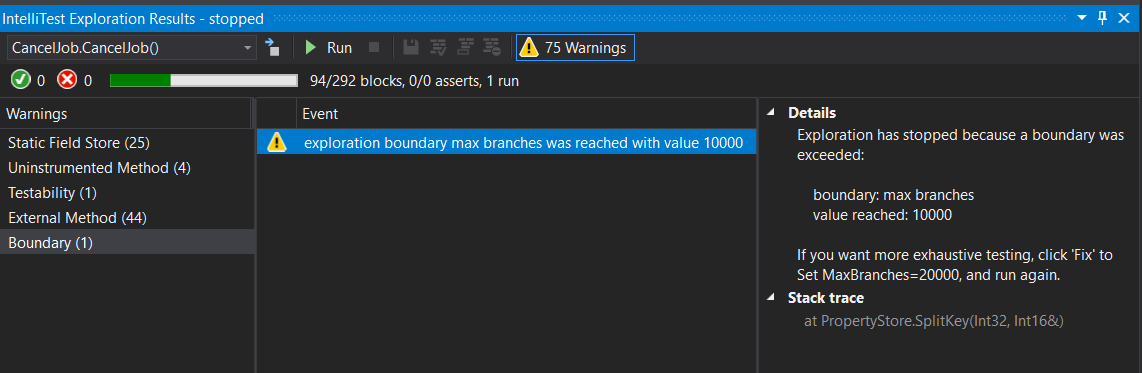


Figure 31 Warning about boundary for branches exceeded

The warning was fixed by set up a new max limit for branches to 20000:

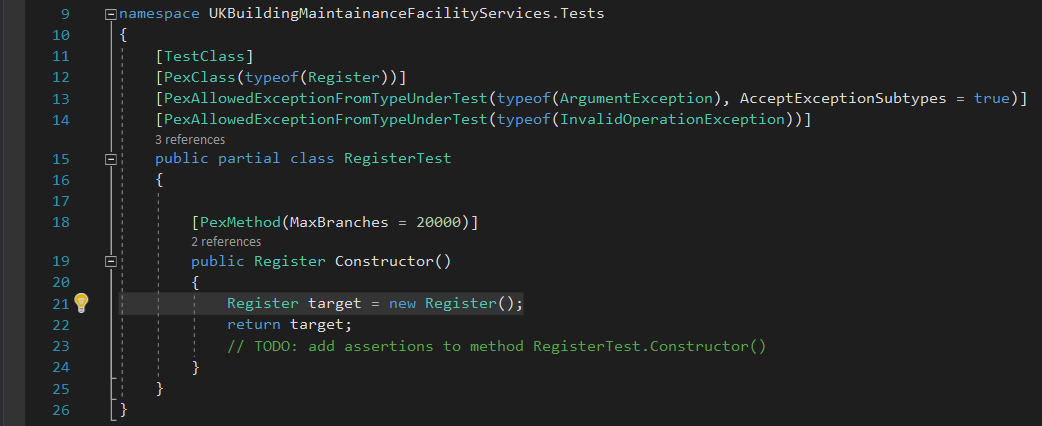


Figure 32 New Boundary fixed for max limit of 20000 branches

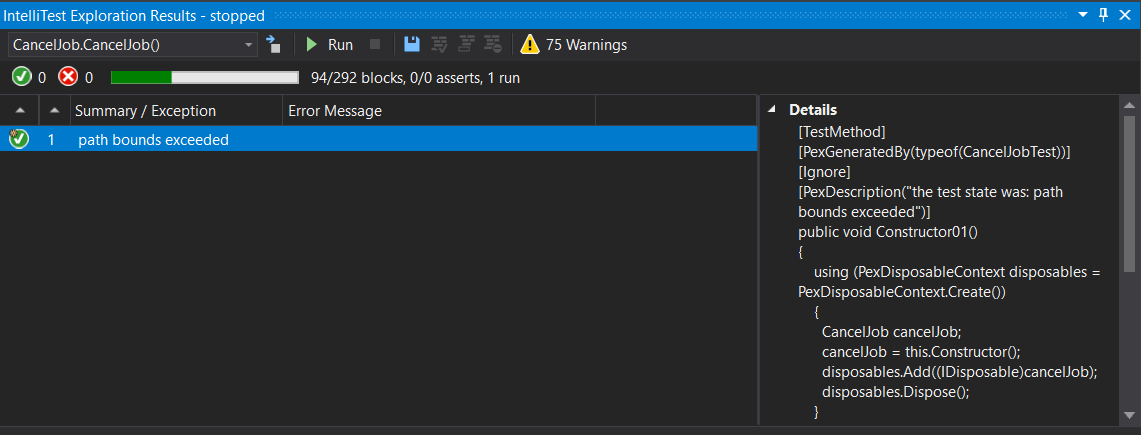


Figure 33 New Boundary fixed for max limit of 20000 branches

### **Black-box Testing**

Table 2 Black-box testing

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Expected Result | Actual Result | Pass or Fail |
| Performance | The response time should be low so that users can access the system fatly | The response time is in the accepted standard limits of accessing the system fatly | Pass |
| Reliability | The system has to be consistent in performance; it means that there should be less or no software failure | The expected consistency of the system has no failure when is running | Pass |
| Security | Authorization techniques should be used to ensure that the stored data in the system is protected from any kind of malicious activity. | Proper techniques of authorization are enabled to protect stored data of users in case of any malicious attack. | Pass |
| Usability | There should be no need of users to make any efforts in accessing the system. The customer service system | The navigation through the whole application is ease of use. | Pass |
| Modifiability | The system should be flexible enough so that relevant changes can be made according to the demands of customers in future | The application can be easy updated with new features for client’s needs. | Pass |

## Test Summary Report

1. **Purpose**

This report describes the different actions executed as part of UK Building Maintenance Services' Testing system.

1. **Application Summary**

‘UK Building Maintenance Services’ system is a desktop application for home or commercial maintenance services. Jobs can be created by the customers through the application for different areas where they can create a job given details of it and can assign an engineer or leave the admin to assign an engineer to be completed. Real-time jobs information is received from *ukservices* database. There are some forms such as Registration, Login, Create Job, Cancel Job, View/Update job status, Select Job, Approve Job, Add Category (specialization of engineer), Authorize/De-authorize Engineer by admin, Customer Feedback and Reports which are combined to fulfil the goal.

1. **Testing Scope**

The scope of testing the application is because all implemented system must meet the user requirements, the functional and non-functional requirements and also, the system requirements.

Out of scope will be the naming violation rule.

1. *In Scope*

Functional Testing for the following modules are in Scope of Testing

* Registration
* Login
* Create Job
* Cancel Job
* View/Update job status
* Select Job
* Approve Job
* Add Category (specialization of engineer)
* Authorize/De-authorize Engineer
* Customer Feedback

1. *Out of Scope*

The naming violations rule error messages during static analysis tests or the dynamic tests will not be retested and will not be used a regression test because the impact will be minor or will not affect the expected results.

Performance Testing was not done for this application.

1. **Types of testing performed**

a) *Smoke Testing* was performed whenever a Build or Rebuild was used within Test context for Testing to ensure that the main functionalities are running accurate, Build can be accepted, and Testing can begin.

b) *System Integration Testing* - This is the Testing conducted on software under test, to check the whole system operates as the expected results. Critical Business outlines were checked to ensure essential functionalities in the software product operates as expected without any failures.

*c) Performance Testing* - this test was conducted to verify the quality of the software product

d) *Security Testing* – this test was done to ensure that the stored data can not be obtained through a malicious attack.

e) Usability Testing – this test was conducted to ensure that the application is easy to use by the users

1. **Test Environment and Tools**

|  |  |
| --- | --- |
| **Application IDE** | Visual Studio |
| **Database** | phpMyAdmin |
| **Tools** | Live Unit Testing, Code Analysis testing |

1. **Recommendations**

* A repeated task done manually every time was time-consuming.
* Many records need to be created for Testing.

1. **Exit Criteria**

* All test cases should be performed – Yes
* All errors in Critical, Major, Medium rigour must be checked and closed – Yes.
* Any open bugs in minor severity – Action plan developed with required dates of closure.

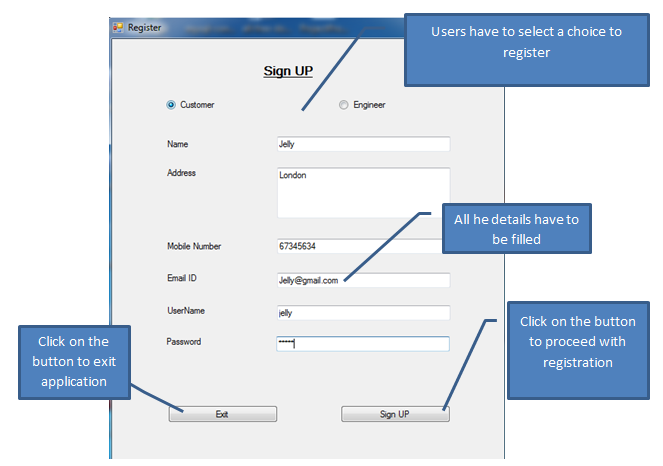
1. **Conclusion/Sign Off**

As the Exit criteria meet and fulfilled as the expected requirements then is advised to be deployed live. Proper User acceptance testing must be completed before ‘Go Live’.

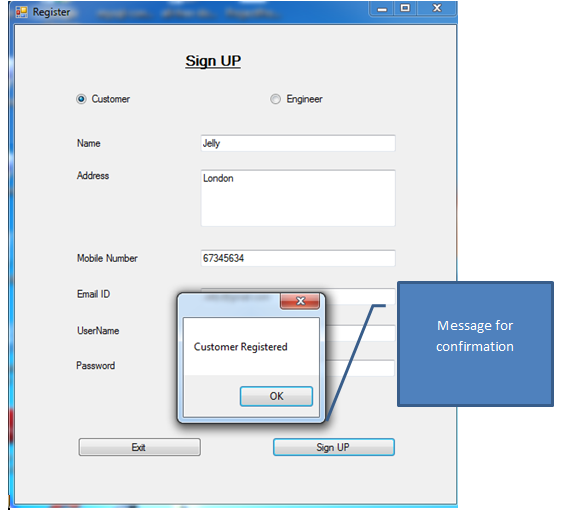
# User manual

User manual provides description of the design for a developed system. It helps users to understand how to perform operations using the new system. User manual gives instructions to the users using a sequential method. Step by step process is described for the users to access the developed desktop application (Hodgson, 2017).

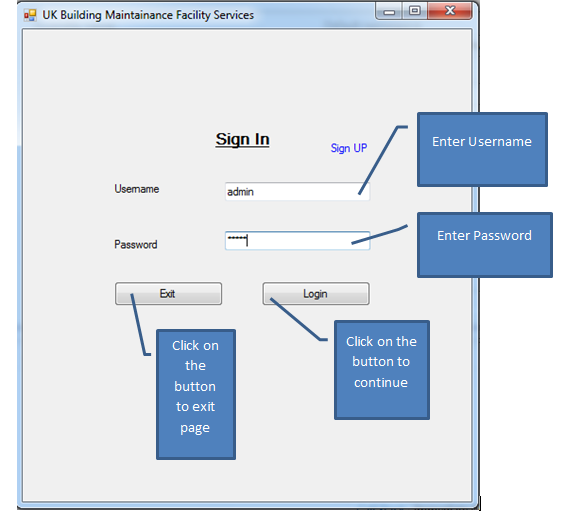
1. First step is that users have to choose an option such as customer or engineer
2. User have to fill in all the information
3. Users can click on exit button to move out of the application
4. After filling all the necessary details, users have to click on ‘sign up’ button to continue with the registration process.

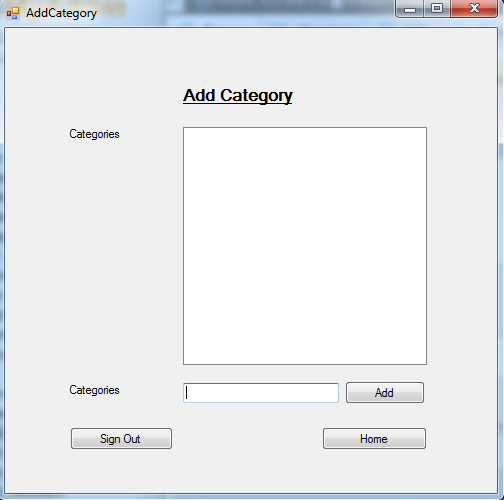


1. Confirmation message will be displayed on the screens of users as shown in the image below.



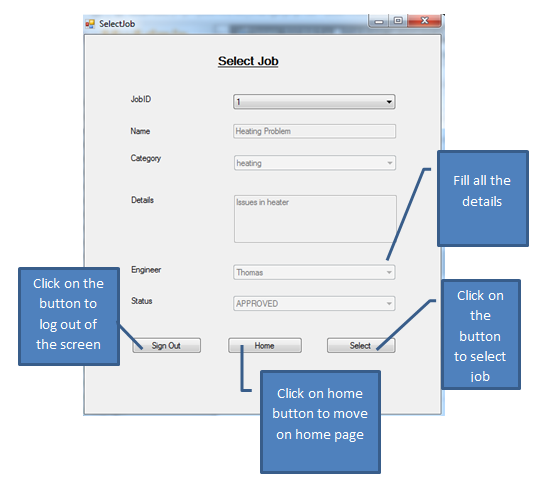
1. Next step is to login to the desktop application



1. Users can move to the option of ‘add category’ after logging in.

Click on the button to continue

1. Engineers can select a job of their type
2. From the drop-down menu, users can select their ID
3. Other details like name, category, details, engineer, status can be selected by a user
4. Users can click on select button to complete the process
5. Navigation button like home or sign out buttons can be clicked to move on different screens of desktop application.



# System Documentation

This type of documentation is designed so that users can view the capabilities, functionalities of a developed system. Beginning from general details to standard information, all this is provided by system documentation. This type of documentation is used to describe details of a product or services that are provided by the system. This system which has been designed using C# is a customer service system in which there are three entities admin, engineer and customer (Devans, 2017)). System documentation for UUK building company is shown in tabular form below:

Table 3: System Documentation

|  |  |  |
| --- | --- | --- |
| System Documentation | | |
| User | System’s Functions | Description |
| Admin | Login | Admin is by default entity which is created to monitor and control the operations performed by other users. Admin has to login first to access the system. |
|  | Authorize engineer | Admin can authorize engineer for a job. This is beneficial to provide security for the system. |
|  | De-authorize engineer | Admin can de-authorize engineer for a job. Once an engineer is de-authorized, system access will be denied for that particular user |
|  | Register engineer/customer | Admin can Register engineer/customer to access the system |
|  | De-register engineer/customer | Admin can De-register engineer/customer to access the system |
|  | Approve job selected by customer | Admin can Approve job selected by customer |
| Engineer | Register | The new engineers have to first sign up to access the system. It is necessary to enter all the details that have been asked in the registration screen |
|  | Login | After successful registration, engineers can login into the system |
|  | Select job | Engineers can select a job from the system as requested by customers |
|  | Update job status | Engineers can update job status by completing details like Engineer ID, name of job, issues and status whether it is started, completed or in progress |
| Customer | Register | New customers have to sign up by entering all the details like name, address, email ID, contact and have to create a password |
|  | Login | Existing customers can directly login to the system |
|  | Select job | Customers can select a job like heating and can write the issue they are facing |
|  | Provide Feedback | For a job in progress or completed, customers can provide their valuable feedback. Customers do not have the authority to provide feedback for a cancelled job by admin |

# Future Recommendations

The customer service system that has been designed for the users of UK building Maintenance Company can make future developments as follows:

* this desktop application as a web application, so admin, engineer and customer can easily access and enter the data.
* Users can access the web application using web browser and can themselves create a user profile and maintain the job details
* It can be made DB on a centralized server to secure the data and to provide easy access
* Mail service and message service for users to know about the progress of operations performed
* Create a log for user operations with the help of DB to maintain history of operations for further reference.

# Conclusion

This is to conclude from this project that software engineering aspects are essential to provide basic guidelines to create a desktop application for different users. This project report has discussed about the user and system requirements for UK Maintenance Company that provides repair and maintenance services to their customers. The system has been designed using C# programming. This report covers the general as well as technical aspects of advance software engineering techniques for the development of desktop application. The developed system will be useful for the users of UK Maintenance Company to maintain the total number of sales on monthly or weekly basis. The system performance will be reliable and also there will be a feature for automatic updates.

# References

Devans, T., 2017. *System Documentation..* [Online]   
Available at: http://www.timothydevans.me.uk/sysdoc.html  
[Accessed 14 01 2018].

Eriksson, U., 2012. *Functional Requirements vs Non-Functional Requirements. [Online] ReQ test..* [Online]   
Available at: https://reqtest.com/requirements-blog/functional-vs-non-functional-requirements/  
[Accessed 10 01 2018].

Guru99, 2017. *Functional Testing Tutorial: What is, Process, Types, & Examples..* [Online]   
Available at: https://www.guru99.com/functional-testing.html  
[Accessed 14 01 2018].

Hodgson, P., 2017. *Tips for writing user manuals..* [Online]   
Available at: https://www.userfocus.co.uk/articles/usermanuals.html  
[Accessed 14 01 2018].

Inflectra, 2016. *What are System Requirements Specifications/Software (SRS)?.* [Online]   
Available at: https://www.inflectra.com/ideas/topic/requirements-definition.aspx  
[Accessed 10 01 2018].

Ofni Systems, 2017. *Functional Requirements (Functional Requirement Specifications, Functional Specs, FRS, FS).* [Online]   
Available at: http://www.ofnisystems.com/services/validation/functional-requirements/  
[Accessed 10 01 2018].

Tutorials Point, 2017. *UML - Class Diagram.* [Online]   
Available at: https://www.tutorialspoint.com/uml/uml\_class\_diagram.htm  
[Accessed 14 01 2018].

Uky, 2017. *Systems implementation..* [Online]   
Available at: http://www.uky.edu/~dsianita/695A&D/lecture5.html [Accessed 27 Dec. 2017]  
[Accessed 14 01 2018].