IN2018: Team Project

IN2018: Team Project

Implementation Report

23/04/2018

Table of Contents

1. Preface

1.1 Purpose of the document

This implementation report describes detailed information of the software architecture of the system, including how the source code is compiled, source files, librares and run-time components. Furthermore, a testing plan is provided, with both functional and non-functional use-cases.

1.2 History of the document

Version	Changes	Responsible for changes	Date
1.0	Added list of source files into document with description.	Henal, Kishan, Sameer, Sai	20/04/2018
1.1	Made changes to functional testing use-cases.	Henal, Kishan, Sameer, Sai	22/04/2018
1.2	Made final changes on the component diagram.	Henal, Kishan, Sameer, Sai	23/04/2018

2. Compilation of source code

2.1 How the system must be compiled and deployed

The source code will be compiled to an output file (byte code). Therefore, all .java files will have their respected .class files which is in the byte code. This code is capable of running in any computer which has a java virtual machine (JVM). These all will be packed and converted to an executable format (.exe) to run on Windows.

2.2 Source files

Source file	**Description**
ActiveDB.java	Class to just to connect database
Bootstrap.java	Central system for theme of entire project
CodeSet.java	Controller for all backend functions
IDgen.java	ID generator
MySQLQueries.java	Controller for all the backend functions which passes MySQL Queries
AddPayment.Java	Frame to add new card for payment
AddTask.java	Creates new tasks
Alert.java	Notification for Office Manager and Shift Manager
BackupRestore.java	Frame to back up and restore the database
CreateJob.java	Create job for the particular customer
CreateTask.java	Add task to the jobs
CustomerSearchList.java	Search customer from the database
CustomerStats.java	Displays statistics from the customer account
Dpayment.java	Payment option for the non-valued customer
Dashboard.java	Dashboard for the Office Manager
Delete.java	Delete for staff and customer accounts
EditCustAcc.java	Edit the customer account details and preferences
EditStaffAcc.java	Edit the staff account details and preferences
EditTask.java	Edit the system tasks
ForgotPassword.java	Frame to request for new password
ICustomerAccount.java	Profile for individual customer account
JobDeadline.java	List of unpaid jobs which has pay due
JobList.java	Frame to display all the jobs
Logger.java	Frame to display logger for office manager
Login.java	Frame to user login
ModifyTask.java	Frame to ask user to whether to modify or delete the task
NewCacc.java	Create new customer account
OCustomerAccount.java	Profile for organisation customer account
Payment.java	Payment for valued customer
PrintLabel.java	Frame to print label
PrintLetter.java	Frame to print reminder letters
PrintReceipt.java	Frame to print receipts
RDashboard.java	Dashboard for receptionist
Restore.java	Asks user for restore confirmation

SDashboard.java	Dashboard for Shift Manager
StaffAccount.java	Staff account profile
StaffSearchList.java	Search for staff
StatusUpdate.java	Update the staff account details/preferences
UpdateTask.java	Updating the status of the task by technician
VaddPayment.java	Add card for valued customers
ResetPass.java	Frame for reset password for requested account

2.3 Programming languages used

Java

Most of the code snips are done in Java:

- Frames (Front end): All of the front end have been creates using frames which uses java.swings and java.awt libraries. It creates two files, .java files, which has all the buttons, other functionalities of the GUI and .form files, which gives the layout for the respective .java files and provides the graphical interface letting components drag and drop while coding.

- Back end: ActiveDB.java file lets the project to connect to the database to JDBC library. MySQLQueries.java file lets us execute the queries which is given in a String format, that passes the String to the database as queries using the connection, created by ActiveDB.java. Bootstrap.java lets us create a standard theme for the entire project. The colour combination and the colour for a particular text (e.g. error messages, general text etc.) is set on the Bootstrap. Finally, the IDgenerator generates the ID for staff, customer and job.

MySQL

- Queries are passed as a String from Java to the database through the connection made by ActiveDB.java file. It will be converted from String to Query and pass the result of the executed query to the particular method in MySQLQueries.java file.

XML

- All the reports are made through XML. The content of the report will be stored in .jrxml file and the report is built with the help of graphical tools provided in .jasper files.
- The parameters in the reports are passed from java to the xml using hashmap in java. The parameters will be mapped using its respected keys and passed through the MySQL queries.

2.4 Created binary files

The entire project is compressed as a .jarfile which can be run in any machine with a java virtual machine (JVM).

2.5 Executables produced

These are produced from our source code so that the compilation can be repeated by a third party.

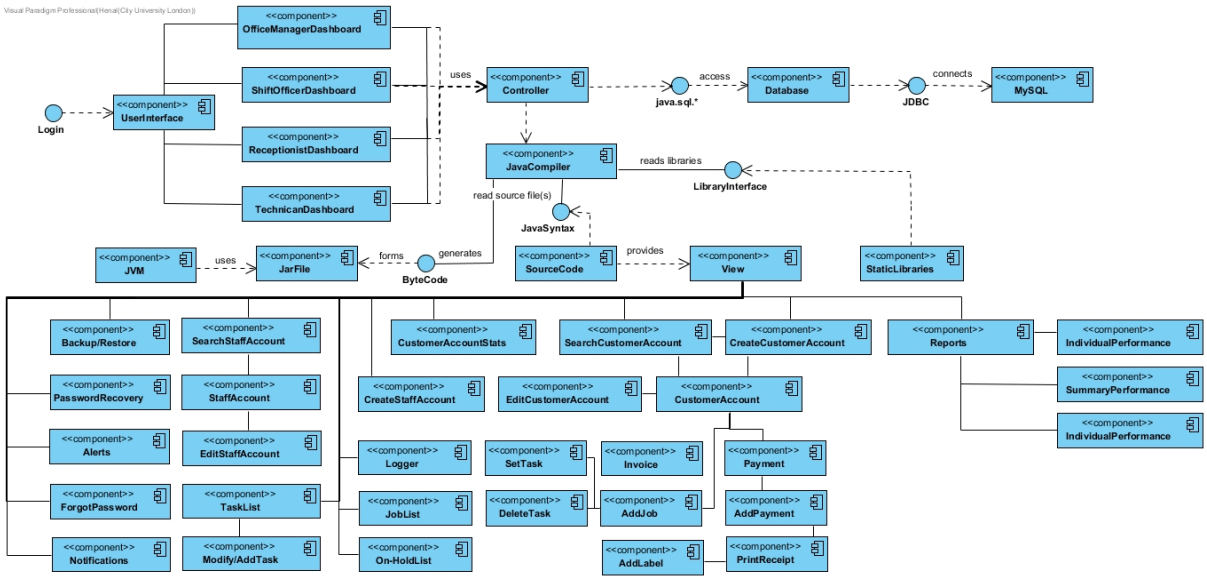
Third party libraries used to compile from the source code:

MySQL_5.1.45
DataChooser
groovy-all
stax-1.2.0.jar
jcommon-1.0.23.jar
castor-xml-1.3.3.jar
core-3.2.1.jar
icu4j-57.1.jar
jakarta-regexp-1.4.jar
lucene-queryparser-4.5.1.jar
commons-beanutils-1.9.3.jar
ecj-4.4.2.jar
jackson-core-2.1.4.jar
jackson-datablind-2.1.4.jar
DataChooser.jar
commons-logging-1.1.1.jar
bcprov-jdk15on-1.52.jar
jackson-annotations-2.1.4.jar
commons-lang-2.6.jar
commons-digester-2.1.jar
lucene-queries-4.5.1.jar
stax-api-1.0.1.jar
lucene-sandbox-4.5.1.jar
javax.inject-1.jar
itext-2.1.7.js6.jar
stax.api-1.0-2.jar
lucene-core-4.5.1.jar
commons-collections-3.2.2.jar
jfreechart-1.0.19.jar
lucene-analyzers-common-4.5.1.jar
olap4j-0.9.7.309-JS-3.jar
jasperreports-6.5.1.jar
castor-core-1.3.3.jar

All the source code (.java files) have their own output file (.class) which is in the byte code. JVN uses this byte code to run on any system.

3. Run-time components

3.1 Component diagram



We are running MySQL Server natively (**not remotely**).

4. **Testing plan -** Use-case testing

Functional testing

Use Case Testing: Login

Use case ID: 1	Use case Name: Login
Test Number: 1	
Objective: Getting access to the BABERS software with different roles.	
Set up: The user is not logged into the system yet.	
Expected results: 1. The details are checked to be correct and is then recorded successfully. 2. The system log stores the results of the login. 3. The user gets access to the dashboard with details based on the privileges.	
Test: 1. The user logs in as the office manager using the account details with the username: 1 and the password: JOQwerty in the text fields provided by the UI. 2. Hit the login button. 3. Account details are checked with the system and the user is logged in successfully. 4. Hit the logout button. 5. The user logs in as the receptionist using the account details with the username: 4 and the password: SCQwerty in the text fields provided by the UI. 6. Hit the login button. 7. Account details are checked with the system and the user is logged in successfully.	
Test record: The system has successfully logged in and has recognised the username and password as correct and redirects the interface to the appropriate dashboard. The log is updated.	
Date: 11 April 2018	Tester: Sameer Ali
Result: Passed	
Date: 13 April 2018	Tester: Kishan Patel
Result: Passed	

Use case ID: 1.2	Use case Name: Login
Test Number: 2	
Objective: Test the alternative flow where the input username or password is incorrect.	
Set up: User's password details do not exist in the system.	
Expected results: 1. The details do not match with the data stored in the system. 2. The system log stores the results of the login for the manager. 3. Error message appears for the user.	
Test: 1. Enter the username: 1 and the password: JSQrery in the text fields provided by the UI. 2. Hit button 'Login'. 3. The system shows a warning message 'Invalid username or password'. 4. The user cannot access the system.	
Test record: Observation meets the expected results; the log is updated.	
Date: 13 April 2018	Tester: Sameer Ali
Result: The dashboard did not display as the office manager couldn't log in.	
Date: 14 April 2018	Tester: Kishan Patel
Result: Passed	

Use case ID: 1.3	Use case Name: Receptionist Login
Test Number: 3	
Objective: Test the alternative flow where the user logs into the wrong dashboard.	
Set up: A office manager logs into the system with the username “” and password “” but is redirected to the wrong dashboard, in this case it's the receptionist login.	
Expected results: 1. The details are checked to be correct and is then recorded successfully. 2. The system log stores the results of the login for the manager. 3. The user gets access but to the wrong dashboard.	
Test: 1. Enter the username: 4 and the password: SCQwerty in the text fields provided by the UI. 2. Hit the login button. 3. The system check and authenticates the details provided and is successfully logged into the system. 4. The user is redirected to the incorrect dashboard.	
Test record: The system has successfully logged in and has recognised the username and password as correct and redirects the interface to the dashboard.	
Date: 13 April 2018	Tester:Sameer Ali
Result: Passed	
Date: 14 April 2018	Tester: Kishan Patel
Result: Passed	

Use Case Testing: Create Customer Account

<table>
<tr><td>Use case ID: 6</td><td>Use case Name: Create Customer Account</td></tr>
<tr><td colspan="2">Test Number: 4</td></tr>
<tr><td colspan="2">Objective: Test the main flow</td></tr>
<tr><td colspan="2">Set up: Authorized personnel can create a customer account by clicking the add customer button in the dashboard, by entering the relevant information about the customer the account is created.</td></tr>
<tr><td colspan="2">Expected results:
1. The customer details are recorded successfully.
2. A customer account is created.</td></tr>
<tr><td colspan="2">Test:
1. An authorised personnel in this case the shift manager logs into the system using the username: 2 and password: SCQwerty
2. When in the dashboard the user can go into the subpanel called customer account and click on create which redirects to the page for creating new customer account.
3. Enter John as the first name, Blair as the last name, 07564899009 as phone number, john.blair@gmail.com as email address, 56 High Street Road as the address, postcode: WCC1 4JP
4. Hit the ‘Submit’ button, and a message stating “Account created.”</td></tr>
<tr><td colspan="2">Test record: Results observed as expected. Customer account successfully created, and the database is updated with new customer details.</td></tr>
<tr><td>Date: 13 April 2018</td><td>Tester: Sameer, Kishan</td></tr>
<tr><td colspan="2">Result: The customer account was created.</td></tr>
<tr><td>Date: 14 April 2018</td><td>Tester: Sameer, Kishan</td></tr>
<tr><td>Result: Passed</td><td></td></tr>
</table>

Use Case Testing: Take Backup

Use case ID: 18	Use case Name: Take Backup
Test Number: 5	
Objective: Test the main flow	
Set up: The office manager is logged into the system and is authorized to take a backup of the entire system when the user deems it necessary.	
Expected results: 1. The system has successfully initiated the backup. 2. A message displays saying, “Backup Successful”. 3. The log entry updates.	
Test: 1. The manager logs in using the username: 1 and password: JOQwerty 2. Redirected to the correct dashboard. 3. On the dashboard clicks on the backup button. 4. Redirects to a popup window with a list of backup already done and click on the backup button to start the process. 5. Message appears stating “Backup successful”	
Test record: Tests were as expected; backup list is updated.	
Date: 13 April 2018	Tester: Kishan Patel
Result: Passed, the backup of the system was successful.	
Date: 14 April 2018	Tester: Sameer Ali
Result: Passed	

Use case ID: 18.2	Use case Name: Take Backup
Test Number: 6	
Objective: Test the alternate flow of the system when there isn't sufficient storage space on the system.	
Set up: The office manager is logged into the system and is authorized to take a backup of the entire system when the user deems it necessary.	
Expected results: 1. The system doesn't create a backup. 2. A message displays saying, "Backup failed". 3. The log entry updates.	
Test: 1. The manager logs in using the username: 1 and password: JOQwerty 2. Redirected to the correct dashboard. 3. On the dashboard clicks on the backup button. 4. Redirects to a popup window with a list of backups already done and click on the backup button to start the process. 5. An error message displays stating that "Backup failed" as there is insufficient storage space.	
Test record: Tests were as expected; backup list is not updated.	
Date: 13 April 2018	Tester: Kishan Patel
Result: Passed, the backup of the system was unsuccessful.	
Date: 14 April 2018	Tester: Sameer Ali
Result: Passed	

Use Case Testing: Create Staff Account

<table>
<tr><td>Use case ID: 22</td><td>Use case Name: Create Staff Account</td></tr>
<tr><td colspan="2">Test Number: 6</td></tr>
<tr><td colspan="2">Objective: Test the main flow of the system to check the addition of creating a new staff account to the BAPERS system.</td></tr>
<tr><td colspan="2">Set up: Office Manager logs into the system to create a new staff account with the correct privileges set for the new user. The office manager will set this up by having inputted the correct details with relevant information pertaining to the new staff user.</td></tr>
<tr><td colspan="2">Expected results:
1. The staff details are recorded successfully.
2. A new staff account (receptionist) is created with the correct access level set.
3. The new staff account is now able to log into the BAPERS system.</td></tr>
<tr><td colspan="2">Test:
1. Office manager logs into the system with the username: 1 and password: JOQwerty.
2. The office manager then hits the button to add new account and then select staff account.
3. Enter the first name: Sam, last name: Watson, phone number: 07654877665, email address: sam.watson@gmail.com and address: 43 North Johnson Street SE2 OU.
4. Enter the username: 10 and password: 1234
5. Select the staff role.
6. Hit the ‘Submit’ button and the new staff account is successfully created.
7. The office manager then logs out of his account and can log in using the newly created staff account with the username: 10 and password: 1234 .
8. It is then redirected to the receptionist dashboard successfully.</td></tr>
<tr><td colspan="2">Test record: Results observed as expected. Staff account successfully created, and the database is updated with new staff details.</td></tr>
<tr><td>Date: 13 April 2018</td><td>Tester: Kishan Patel</td></tr>
<tr><td colspan="2">Result: The staff account was successfully created.</td></tr>
<tr><td>Date: 14 April 2018</td><td>Tester: Sameer Ali</td></tr>
<tr><td>Result: Passed</td><td></td></tr>
</table>

Use case ID: 22.2	Use case Name: Create Staff Account
Test Number: 7	
Objective: Testing the alternate flow where the new account being created is already in the BAPERS system.	
Set up: The office manager logs in to the system with his details, and then tries to make a new staff account but as the details already exists within the system it fails.	
Expected results: 1. The office manager is not able to create the staff account as it already exists within the system. 2. Gets an error stating the account already exist.	
Test: 1. Office manager logs into the system with the username: 1 and password: JOQwerty. 2. The office manager then hits the button to add new account and then select staff account. 3. Enter the first name: Sam, last name: Watson, phone number: 07654877665, email address: sam.watson@gmail.com and address: 43 North Johnson Street SE2 OU. 4. Enter the username: 10 and password: 1234 5. Select the staff role. Hit the 'Submit' button but gets an error stating "Account already exists".	
Test record: Results are as expected, observed the error present on the dashboard as account already exists.	
Date: 13 April 2018	Tester: Kishan Patel
Result: Passed	
Date: 15 April 2018	Tester: Sameer Ali
Result: Passed	

Use Case Testing: Initiate Account Suspension

Use case ID: 32	Use case Name: Initiate Account Suspension
Test Number: 8	
Objective: Test the main flow	
Set up: The office manager can use the dashboard to change the customer account status by using the search feature in the dashboard to first find the customer and then by clicking it being able to edit the details.	
Expected results: 1. The customer account is set to suspend. 2. The database updates to store the additional information.	
Test: 1. Search customer through the dashboard by entering the name: John. 2. Click on the result found by the search feature and get redirected to the edit customer details page. 3. Click 'Account Status' and this in turn sets the account as suspended. 4. A red bar goes across the account status from green to indicate the suspension.	
Test record: Test results as expected, customer account status was set to suspend.	
Date: 13 April 2018	Tester: Sameer, Kishan
Result: The log entry was updated with the further details.	
Date: 14 April 2018	Tester: Sameer, Kishan
Result: Passed	

Use Case Testing: Record Completed Job

Use case ID: 44	Use case Name: Record Completed Job
Test Number: 9	
Objective: Test the main flow	
Set up: After the job has been completed there is an option to change the status from pending to completed which will then be automatically updated in the database. By using the dashboard, we can search the job list and use the search option to narrow down the and once found use that to change the status of the job.	
Expected results: 1. The technician can find the job from the database and change the status to completed. 2. The job status is recorded as completed by the system.	
Test: 1. The job is completed, the technician goes on to the job list. 2. By inputting the job id into the search feature, it can narrow down the search to find the exact job. 3. By clicking on the specific job can change the status from pending to completed. 4. Hit the confirmation button, the system should respond by saying status has been changed.	
Test record: The observation was consistent with the expected results.	
Date: 13 April 2018	Tester: Sameer Ali
Result: The status has changed from pending to completed, and the database is updated.	
Date: 14 April 2018	Tester: Kishan Patel
Result: Passed	

Use Case Testing: Search Staff

Use case ID: 25	Use case Name: Search Staff
Test Number: 10	
Objective: Test the main flow	
Set up: By using the dashboard the Office Manager can use the UI feature of Search Staff by entering the keyword for the search result. The staff ID should come up with a match with results.	
Expected results: 1. The staff details are in the database. 2. By searching the staff ID, the system can find a match. 3. The system retrieves the information from the database and displays it on the dashboard.	
Test: 1. Using the dashboard and going into the search staff feature enter the keyword to find a specific match. 2. Entering the Staff ID '10' or the staff name 'Sam' 3. Hit the search button, the system should respond by displaying the search result found.	
Test record: The result was consistent with the data inputted.	
Date: 13 April 2018	Tester: Sameer, Kishan
Result: Search result has resulted in a match from the database.	
Date: 14 April 2018	Tester: Sameer, Kishan
Result: Passed	

Use case ID: 25.2	Use case Name: Search Staff
Test Number: 11	
Objective: Testing the alternate flow where the inputted staff id is unknown and cannot be found in the database.	
Set up: By using the dashboard the Office Manager can use the UI feature of Search Staff by entering the keyword for the search result. The Staff ID should come up as unmatched.	
Expected results: 1. The staff ID is NOT recorded in the database. 2. No search result is found with the specific data inputted.	
Test: 1. Enter the Staff ID: 11 and hit the search button. 2. The system should not show any results as there is no such data in the database to be found. 3. Hit button 'Cancel' to go back to the main page of the dashboard.	
Test record: The expected result was consistent with the data inputted.	
Date: 13 April 2018	Tester: Sameer, Kishan
Result: The staff id was not retrieved, and a blank was displayed.	
Date: 14 April 2018	Tester: Sameer, Kishan
Result: Passed	

Use Case Testing: Reports

Use case ID: 12	Use case Name: Reports
Test Number: 12	
Objective: Test the main flow	
Set up: Using the dashboard an office manager can search for a specific customer report by entering their name into the search bar or using the job sheet id under the Reports section of the dashboard.	
Expected results: 1. The customer name should be found in the system. 2. The customer report is generated by the system.	
Test: 1. The office manager logs into the system using the username: 1 and password: JOQwerty 2. Go into the subpanel customer account and in the search field enter the name of customer: John Blair. 3. Enter the customer name 'John Blair' into the search field of the dashboard. 4. Shows a match in the database for the following search. 5. Click on the customer name which redirects to the customer page. 6. Select the list of jobs and click generate to generate a report.	
Test record: Observation meets the expected results, log updated.	
Date: 13 April 2018	Tester: Sameer, Kishan
Result: The system generated a report for the specified customer.	
Date: 14 April 2018	Tester: Sameer, Kishan
Result: Passed	

Use case ID: 12.2	Use case Name: Reports
Test Number: 13	
Objective: Test the alternate flow in which no such customer exists in the database.	
Set up: search for a customer report by entering the name 'Will' into the search bar or using the job sheet id under the Reports section of the dashboard, but the system should not allow a report to be generated as the customer 'Will' doesn't exist.	
Expected results: 1. The customer name/job id does NOT exist in the database. 2. The customer report is not generated by the system.	
Test: 1. Enter 'Will' in the search field, press the search button. 2. The system shows a blank list indicating that no such customer exists. 3. Hit the 'x' mark on the top right-hand side corner to return to the previous page	
Test record: The result was consistent with the data inputted.	
Date: 13 April 2018	Tester: Sameer Ali
Result: The search result did not result in a match and system didn't generate report.	
Date: 14 April 2018	Tester: Kishan Patel
Result: Passed	

Use Case Testing: Assign Job

Use case ID: 47	Use case Name: Assign Job
Test Number: 14	
Objective: Test the main flow for an existing technician that has no previous jobs pending.	
Set up: Technician logs into the system using the username and password, the system checks to see that there are no pending jobs for the particular user.	
Expected results: 1. System is checked, and a new job is assigned to the technician. 2. Job is displayed on the dashboard under the work details.	
Test: 1. Enter the username: 5 and password: KPQwerty for the technician that has no pending jobs from previous days. 2. Hit the login button, the system scans the database for any jobs that haven't been completed. 3. As no pending jobs the system assigns new job to the user. 4. The dashboard is updated with the job id.	
Test record: Results was as expected; new job is assigned to the user.	
Date: 15 April 2018	Tester: Kishan Patel
Result: Passed	
Date: 16 April 2018	Tester: Sameer Ali
Result: Passed	

Use case ID: 47.2	Use case Name: Assign Job
Test Number: 15	
Objective: Test the alternate flow of an existing technician that has a pending job from previous day.	
Set up: Technician logs in with the username and password, the system check to see if any jobs have been assigned previously to the user to be completed.	
Expected results: 1. Checks the database and assigns pending jobs to the technician. 2. Job is displayed on the dashboard under the Work Details.	
Test: 1. Enter the username: 5 and password: KPQwerty for the technician that has not completed his job on the previous day. 2. Hit the login button, the system scans the database for any jobs that haven't been completed by the user and displays it to the dashboard.	
Test record: Expected results observed, jobs are shown correctly from the data.	
Date: 15 April 2018	Tester: Sameer, Kishan
Result: The data didn’t show one of the jobs that were pending.	
Date: 16 April 2018	Tester: Sameer, Kishan
Result: Passed	

Use case ID: 47.3	Use case Name: Assign Job
Test Number: 16	
Objective: Test the alternate flow when a new technician logs in.	
Set up: Technician logs in with the username and password, the system check to see if any jobs have been assigned previously to the user to be completed.	
Expected results: 1. Checks the database and should not assign pending jobs to the technician. 2. No job is displayed on the dashboard under the Work Details. 3. The system will assign a new job to the technician as there is no previous records. 4. The Work Details update to show new jobs.	
Test: 1. Enter the username: 14 and password: IOQwerty for the new technician. 2. Hit the login button, the system scans the database for any previous jobs for the technician. 3. Technician views the updated dashboard.	
Test record: Expected results observed, a new job is assigned to the user.	
Date: 15 April 2018	Tester: Sameer, Kishan
Result: New job assigned from the job list in the database.	
Date: 16 April 2018	Tester: Sameer, Kishan
Result: Passed	

Use Case Testing: Assign Department

Use case ID: 48	Use case Name: Assign Department
Test Number: 17	
Objective: Test the main flow	
Set up: Receptionist assigns a new job to the Copy Room department. Technician logs in with the username and password that works in the Copy Room.	
Expected results: 1. The system assigns the job to the technician at the designated department. 2. The job is recorded as In-Progress. 3. The job is recorded in the log.	
Test: 1. Enter the username: 14 and password: IOQwerty for the technician that works in the Copy Room. 2. Hit the login button, the system scans the database for any jobs for the specific department. 3. Assigns the jobs to the dashboard of the technician.	
Test record: Expected results observed, jobs are shown correctly from the data.	
Date: 15 April 2018	Tester: Sameer, Kishan
Result: The data didn't show one of the jobs that were pending.	
Date: 16 April 2018	Tester: Sameer, Kishan
Result: Passed	

Non-functional testing

For the production version of BAPERS, we consider testing Security and Scalability as non-functional requirements.

Security

Whenever a user attempts to log in with an incorrect password, it will give an error message stating “Invalid username or password”. It also facilitates user to reset their password. In all these cases, the Office Manager has the access to view the logger. The logger holds a record of all activities done by all the accounts, including the Office Manager itself.

If any account is created for a customer or staff, the record is held. Similarly, if anyone creates a new job or makes a payment, it also holds a record of it. It holds a record of the name of the activity, time of the activity which has taken place and from which account number the activity has processed.

Scalability

The system is capable of handling even a huge data load and it sorts based on the priority but it is still capable of delivering services for both standard and priority customer without missing any deadlines even if the data load is high. It sorts by the deadlines and assigns tasks to the technicians of the respected department. In that way, even if the data load is high it delivers the service on or before the deadline. This type of approach helps to handle any amount of data irrespective of its size. We created our own algorithm which bypasses the data access conflict such as assigning a single task to multiple technicians. The designed algorithm works efficiently regardless of the data loads.

Security

Requirements ID:	1	**Requirements Type:**	**NFR**	**Event / Use Case #**	#1
Description:	The BAPERS system shall only allow users to sign in if they input the correct username and password.				
Rationale:	This provides high security as the system is only accessible by permitted users and prevents unauthorized access to the system.				
Source:	Interview with the Mr Lancaster.				
Fit Criteria:	The system will verify if the log in details that the user provides are correct, allowing them to gain access to the dashboard. If the details are incorrect, the user will be presented with an error message preventing them to gain access to the dashboard. The user is also given the option to reset their password if necessary.				
Customer Satisfaction:	5	**Customer Dissatisfaction:**		5	
Priority:	An essential requirement	**Conflicts**		None	
Supporting Material:	Source code			Volere Source: BAPERS	
History:	A new requirement				

Scalability

Requirements ID:	2	**Requirements Type:**	**NFR**	**Event / Use Case #**	#47
Description:	The BAPERS system should assign jobs to technicians based on deadlines and department.				
Rationale:	This provides high scalability in the case that the data load is high as the tasks are delivered by their deadlines, which prevents multiple technicians from being assigned the same task, allowing the data to be handled regardless of its size.				
Source:	BAPERS brief.				
Fit Criteria:	The system efficiently sorts and assigns job tasks through their deadlines to technicians of the respective departments. Technicians will be provided with information of a new task when they complete their assigned task.				
Customer Satisfaction:	5	**Customer Dissatisfaction:**	5		
Priority:	Essential	**Conflicts**	None		
Supporting Material:	Source code		Volere Source: BAPERS		
History:	A new requirement				