Test Case Template and Final Contribution Statement (CS360)

Issue Logging System for Allied Bank Limited

Group Number: 24
Aadam Nadeem
Maleeha Masood
Malik Ali Hussain
Muhammad Raheem Zafar
Shahrukh Kemall

Course: Software Engineering CS360
Instructor: Suleman Shahid
University: Lahore University of Management Sciences (LUMS)

Version: 1.0
Date: (20/05/2020)
Number of hours spent on this document: 75

May 21, 2020

Revision History

Date	Version	Description	Author
20th May, 2020	1.0	Initial Draft of Test Cases	Group 24

Table of Contents

Work Division plan

Please share the work plan for each member.

Member name	Work done (**Clearly** write the name of sections, test case numbers, and the other ways you contributed)	Approximate Equal contribution to the group work (Yes/No) + Remarks if any
Aadam Nadeem	Section 3 + summary +API testing secondary+ Section 1.5	Yes
Maleeha Masood	Initial Formatting of the Document Sections 1.1, 1.2, 1.3, 1.4, 1.6 Section 2.2 Sections 3.1 to 3.11 + summary Section 5	Yes
Malik Ali Hussain	Sections 3 + summary + API testing secondary	Yes
Muhammad Raheem Zafar	Overall formatting + summary + API testing Secondary	Yes
Shahrukh Kemall	API testing Primary + summary 3.31-3.36	Yes

Test Cases

1. Introduction - Maleeha Masood and Aadam Nadeem

1.1 Document Purpose

The purpose of this document is to enlighten its reader with the methodology of the various kinds of testing that have been done on our developed mobile application- ILS. This document also gives its reader an idea about the level of rigor used in testing the application as well as the degree of robustness of the application against invalid inputs and scenarios. It aims to help out individuals in the future who wish to test our application by laying the foundations of test cases already done.
Within this document, the reader can find a brief introduction of the document and any pre-requisites needed, followed by manual testing cases done on the application and then, a description of the automated testing tools used. Wherever testing is done, the test data used, and the results of the tests are also made available for the reader to keep them aware of the complete scenario.

1.2 Document Scope

The scope of this document is describing that software testing done on the ILS app. Both functional and non-functional testing have been done on the app. Both validation and to an extent, defect testing has been done for each test case by entering in valid and invalid inputs. All types of unit, sub system integration and system tests have been performed keeping the initially desired functionalities of the application in mind. About 70% of the test cases are done using black box testing. The remaining either use white box or grey box testing. Primarily, dynamic testing has been done in the test cases to ensure that the flow of the application is non problematic. Majority of our tests are manual; however, we have used automated testing tools too.
Testing is done mostly on the User Interface since it is the primary mode of interaction between the application and the user and so is the platform that users interact with the most. Testing has also been done on the backend database since it is an integral part of the correct functioning of the mobile application. We have also done Interface Testing: test cases that determine the correctness of the functionality of the API have been executed so that the interaction between the UI and the backend can be tested for errors and anomalies.

1.3 Definitions, Acronyms and Abbreviations

❑ ABL: Allied Bank Limited

❑ Android Studio: The official integrated development environment for Google’s Android operating system.

❑ API: Application Programming Interface- An application programming interface is a computing interface which defines interactions between multiple software intermediaries.

❑ APK: Android Package

❑ App: Application

❑ Emulator: An emulator is hardware or software that enables one computer system (called the host) to behave like another computer system (called the guest). An emulator typically enables the host system to run software or use peripheral devices designed for the guest system.

❑ ID: Identification

❑ ILS: Issue Logging System

❑ RQ: Requirement

❑ SDS: Software Design Specification Document

❑ SRS: Software Requirements Specification Document

❑ SQL Injection Attack: In an SQL injection attack, an attacker supplies an untrusted SQL query modifying input to a program. This input gets processed by an interpreter as part of a command or query. In turn, this alters the execution of that program.

❑ UI: User Interface

❑ Widget Testing: A widget test (in other UI frameworks referred to as component test) tests a single widget. The goal of a widget test is to verify that the widget's UI looks and interacts as expected.

❑ XSS Attack: a web security attack caused by inserting html or script files into placeholders that may get executed if preventions are not in place.

1.4 References

APK of the app:
https://drive.google.com/file/d/1jbk_5rqy5c_g2txvWIynOVLYTcGDOjkw/view?usp=sharing

Repository of the ILS Project:
https://github.com/maleehamasood/LUMS-CS360-Project

SRS of the app:
https://github.com/maleehamasood/LUMS-CS360-Project/blob/master/Group24_SRS_S20.pdf

SDS of the app:
https://github.com/maleehamasood/LUMS-CS360-Project/blob/master/Group24_SDS_S20.pdf

1.5 Overview of Use Cases Used

The use cases used in this document are based on our specified functional and non-functional requirements, listed in the requirements document. While a wide variety of use cases could be explained, explained in this document are a few from the general requirements and use cases given to us by our client, to add diversity to the document. All types of unit, sub system integration and system tests have been performed keeping the initially desired functionalities of the application in mind. Some prominent examples of use cases used in this document are log in the app, log in an issue, view status of issue, change role of an employee and assign open issues, of which some are restricted to a specific precondition in the application.

1.6 Prerequisites to Execute Test Cases

The main pre-requisite for executing the manual test cases is are a stable Wi-Fi connection and either a downloaded and installed copy of the apk of the app on an Android phone or an Android Emulator. For the automated test cases, Android Studio as well as the automated testing tool used along with the apk of the app are needed. For testing with the flutter testing package, VS Code is preferred to run the tests.

A copy of the SRS and the SDS must also be accessible to reader to map the use cases, functional and non-functional requirements references in this document. The use cases can be mapped onto those mentioned in Section 3.3.3 of the SDS. The functional requirements can be mapped onto those mentioned in Section 1 of the SDS. The non-functional requirements can be mapped onto those mentioned in Section 4 of the SRS.

2. Testing Environments

2.1 Environment used by member 1 Aadam Nadeem

Machine Name	HP-ENVY	DB Directory	mysql://b46b3837d6e6e2:beefca2d@us-cdbr-iron-east-01.cleardb.net/heroku_43fced943439ae3?reconnect=true		
Machine Specs	16GB RAM, 2.3GHz Intel Core-i7	DB	Clear DB	Client Server /Back-End	000webhostapp URL: https://ilslumsapp.000webhostapp.com
Device name	Samsung Galaxy S8	Device specs	Android version 9		
Tester Name	Aadam Nadeem	Test Date	20th May 2020		

2.2 Environment used by member 2 Maleeha Masood

Machine Name	MacBook	DB Directory	mysql://b46b3837d6e6e2:beefca2d@us-cdbr-iron-east-01.cleardb.net/heroku_43fced943439ae3?reconnect=true		
Machine Specs	8GB RAM, 1.2 GHz Dual-Core Intel Core M	DB	ClearDB	Client Server /Back-End	000webhostapp URL: https://ilslumsapp.000webhostapp.com
Device 1 name	Samsung Galaxy Note 8	Device 1 specs	Android Version 9, One UI Version 1.0		
Device 2 name	Samsung Galaxy S9	Device 2 specs	Android Version 9, One UI Version 1.0		
Tester Name	Maleeha Masood	Test Date	18th May 2020		

2.3 Environment used by member 3 Malik Ali Hussain

Machine Name	HP-ENVY	DB Directory	mysql://b46b3837d6e6e2:beefca2d@us-cdbr-iron-east-01.cleardb.net/heroku_43fced943439ae3?reconnect=true		
Machine Specs	8GB RAM, 2.4GHz Intel Core-i5-6200U	DB	ClearDB	Client Server /Back-End	000webhostapp URL: https://ilslumsapp.000webhostapp.com
Device name	Vivo S1		Device specs	Android version 9, 4GB RAM, 2GHz Octa-core.	
Tester Name	Malik Ali Hussain		Test Date	20th May 2020	

2.4 Environment used by member 4 Raheem Zafar

Machine Name	HP-Notebook	DB Directory	mysql://b46b3837d6e6e2:beefca2d@us-cdbr-iron-east-01.cleardb.net/heroku_43fced943439ae3?reconnect=true		
Version	8GB RAM, 2.4GHz Intel Core-i3-6200U	DB	Clear Db	Client Server /Back-End	000webhostapp URL: https://ilslumsapp.000webhostapp.com
Device name	Samsung Galaxy S7		Device specs	Android version 8.0.0	
Tester Name	Muhammad Raheem Zafar		Test Date	19th May 2020	

2.5 Environment used by member 5 Shahrukh Kemall

Machine Name	MacBook Pro	DB Directory	mysql://b46b3837d6e6e2:beefca2d@us-cdbr-iron-east-01.cleardb.net/heroku_43fced943439ae3?reconnect=true		
Machine Specs	8 GB RAM, 2.7GHz Dual Core, Intel Core i5.	DB	Clear Db	Client Server /Back-End	000webhostapp URL: https://ilslumsapp.000webhostapp.com
Device name	Oppo F5Youth		Device specs	Android version 7.1.1	
Tester Name	Shahrukh Kemall		Test Date	20th May 2020	

3. Test Cases

3.1 List of test cases

Testcase ID	Use Case ID	Functional RQ ID	Non-Functional RQ ID	Testcase name	Performed by	Successful/Failed
Module: Login System						
T0001	1	1	-	Wi-Fi connectivity	Maleeha Masood	Failed
T0002	1	1	-	Login Credentials	Maleeha Masood	Successful
T0003	1	1	-	Copying Password and Pasting it in the Password Field	Maleeha Masood	Failed
T0004	1	1	4.2.2	Login Screen and XSS Attack	Maleeha Masood	Successful
T0005	1	1	4.2.2	Login Screen and SQL Injection Attack	Maleeha Masood	Successful
T0006	1	1	-	Transition from Login to Menu Screen for All Roles	Maleeha Masood	Successful
Module: Main and Specific Menu						
T0007	1		-	Menu and Specific Menu screens	Aadam Nadeem	Successful
T0008	1		-	Banner Image in Menu and Specific Menu screens	Aadam Nadeem	Failed
T0009	-		-	Going back from menu screens	Aadam Nadeem	Failed
Module: Info Screens						
T0010	-	-	-	View info of function in use	Aadam Nadeem	Successful
Module: Log an Issue						
T0011	2	-	-	Update Category and Sub-Category lists according to the Area and Category Selection	Malik Ali Hussain	Successful

T0012	2	4	-	Log an Issue on behalf of any other employee	Malik Ali Hussain	Successful
T0013	2	3	-	Mandatory Fields must be filled to Log an Issue Successfully	Malik Ali Hussain	Successful
T0014	2	-	-	User must be informed whether Issue was logged successfully or not	Malik Ali Hussain	Successful
T0015	2	22	-	Email is sent to the Initiator and relevant support team upon logging an issue	Malik Ali Hussain	Successful
Module: Fetch Lists from the DB according to the given scenario						
T0016	3 and 4	5,7 and 13	-	Fetch Lists from DB.	Raheem Zafar	Successful
T0017	-	-	-	Latest Issue on top.	Raheem Zafar	Successful
Module: View your Issues						
T0018	3	6	-	Update Status of an Issue – PopUp	Raheem Zafar	Successful
T0019	3	6	-	Update Status of an Issue	Raheem Zafar	Successful
T0020	3	24 and 28	-	Email confirmation of issue status update.	Raheem Zafar	Successful
Module: Check all Issues						
T0021	4	7 and 8	-	Check all Issues – Detail Screen	Raheem Zafar	Successful
T0022	4	7 and 8	-	Take up an Issue	Raheem Zafar	Successful
T0023	4	-	-	Email confirmation of issue taken up.	Raheem Zafar	Successful
Module: Issues taken up						
T0024	-	9 and 10	-	Issues Taken up – Detail Screen	Raheem Zafar	Successful
T0025	-	9 and 10	-	Issues Taken up – Action Commit (Mark as Open and Mark as Fixed)	Raheem Zafar	Successful
T0026	-	17		List employees to reassign	Aadam Nadeem	Successful

				issues to		
T0027	-	-	-	Search for employees to reassign issue to.	Aadam Nadeem	Successful
T0028	-	-	-	SQL injections in Search for employees to reassign issue to.	Aadam Nadeem	Failed
T0029	-	17	-	Reassign an issue to an employee	Aadam Nadeem	Successful
T0030				Email Confirmation upon reassigning an issue	Aadam Nadeem	Successful
T0031				User must be informed whether Issue was reassigned successfully or not	Aadam Nadeem	Successful
Module: Manager Team Issues						
T0032	-	13	-	Manager Team Issues – Detail Screen	Shahrukh Kemall	Successful
T0033	-	7 and 8	-	Reassign Issue -Button	Shahrukh Kemall	Successful
T0034	-	16 and 17	-	Reassign Issue – Employee Detail Screen	Shahrukh Kemall	Successful
T0035	-	16 and 17	-	Reassign Issue – By List	Shahrukh Kemall	Successful
T0036	-	16 and 17	-	Reassign Issue- Search Employee PopUp and Detail Screen	Shahrukh Kemall	Failed
T0037	-	16 and 17		Email confirmation of Re-assigned issue.	Shahrukh Kemall	Successful
Module: Changing the Attribute of an Employee						
T0038	5	20 and 21	-	Employee list is displayed to Admin when changing an employee's attribute	Malik Ali Hussain	Successful
T0039	5	20 and 21	-	Directly search an employee from the database for changing the attribute	Malik Ali Hussain	Failed
T0040	5	20 and 21	-	Admin is able to see the details of the employee before	Malik Ali Hussain	Successful

				changing the employee’s attribute		
T0041	5	20 and 21	-	New Attribute must be selected before updating the employee details	Malik Ali Hussain	Successful
T0042	5	20 and 21	-	Admin must be informed whether the attribute was changed successfully or not	Malik Ali Hussain	Successful
Module: Generate Reports						
T0043	-	18 and 19	-	Generate Report for Manager and Monitor Roles	Maleeha Masood	Successful
T0044	-	18 and 19	-	Graphs Fetching Data Real-Time as Database Gets Updated	Maleeha Masood	Failed
T0045	-	18 and 19	-	Back Button of the Multiple Graphs of Generate Report Screen	Maleeha Masood	Successful
Module: Logout						
T0046	-	2	-	Logout from Any Role	Maleeha Masood	Successful

3.2 Test Case T0001: Wi-Fi connectivity – Maleeha Masood

3.2.1 Description

On the login screen page, users are prompted to enter in their already known login credentials. Once entered, these credentials are matched with the one's stored in the Database by querying the database using the entered employee ID. Wi-Fi availability is necessary to connect to the database using the API and thus, for proper functionality.

3.2.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection that can be switched off, a valid employee ID and password pair.

3.2.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Turn the Wi-Fi of the device on.		Wi-Fi on.		Successful	
2	Launch the ILS application.		Application launched.		Successful	
3	Enter a valid employee ID. Enter a valid password. Press Login button.	21100217 abc21100217	Show Loading Dialog Box.		Successful	
4	Turn the Wi-Fi off on the device.		Wi-Fi off.			
5	Repeat step 2-3		Show Error message of no Wi-Fi available.	Showed Invalid Credentials Entered Dialog Box.	Failed	001
			Test Case Status	Failed		

3.3 Test Case T0002: Login Credentials – Maleeha Masood

3.3.1 Description

On the login screen page, users are prompted to enter in their already known login credentials. Once entered, these credentials must match those stored in the database to proceed forward.

3.3.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator and a Wi-Fi connection.

3.3.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Launch the ILS application.		Application launched.		Successful	
2	Enter a valid employee ID. Enter a valid password. Press Login button.	21100217 abc21100217	Show Loading Dialog Box.		Successful	
3	Repeat steps 1-2 with the different values as in 3.3.4.				Successful	
			Test Case Status	Successful		

3.3.4 Array of values

Array of values					
	Scenario 2	**Scenario 3**	**Scenario4**	**Scenario 5**	**Scenario 6**
Value1	21100217	abc21100217 (letters are not allowed in employee ID format)	2110021	21100217	21100217
Value2	abc	abc21100217	abc21100217	abc2110021	abc21100190 (correct password of another employee ID)
Expected Result	Password must be upto 6 characters error message.	Employee ID is not valid error message.	Invalid Credentials Entered Dialog Box.	Invalid Credentials Entered Dialog Box.	Invalid Credentials Entered Dialog Box.
Actual Result (if different from expected)					
Successful/Failed	Success	Success	Success	Success	Success
Environment Nbr (if failed)					
Log Number (if failed)					

3.4 Test Case T0003: Copying Password and Pasting it in the Password Field – Maleeha Masood

3.4.1 Description

On the login screen page, users are prompted to enter in their already known login credentials. To avoid a scenario where attacker can utilize a copied valid password from the clipboard, copying and pasting of passwords in the password field should not be allowed.

3.4.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection and a valid employee ID and password pair.

3.4.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Launch the ILS application.		Application launched.		Successful	
2	Enter a valid employee ID. Enter a valid password.	21100217 abc21100217			Successful	
3	Copy the password and empty the password field.					
4	Paste the copied password.		Password Field should remain empty.	Copied Password Pasted.	Failed	
5	Press Login Button.		Password must be up to 6 characters error message.	Showed Loading Dialog Box.	Failed	002
			Test Case Status	Failed		

3.5 Test Case T0004: Login Screen and XSS Attack – Maleeha Masood

3.5.1 Description

On the login screen page, users are prompted to enter in their already known login credentials. To avoid a scenario where attacker can cause an XSS attack by using script tags, a user should not be allowed to enter scripts in the employee ID field.

3.5.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator and a Wi-Fi connection.

3.5.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Launch the ILS application.		Application launched.		Successful	
2	Enter a script in the employee ID field. Enter a valid password.	<script>alert('HI')</script> abc21100217			Successful	
3	Press Login Button.		Employee ID is not valid error message.		Successful	
			Test Case Status	Successful		

3.6 Test Case T0005: Login Screen and SQL Injection Attacks – Maleeha Masood

3.6.1 Description

On the login screen page, users are prompted to enter in their already known login credentials. To avoid a scenario where attacker can cause an SQL injection attack, a user should not be allowed to enter special characters like ‘ or – (SQL comment) or ; (to start a second, new statement in SQL) and others in the employee ID field since it is the input that is used to query the database.

3.6.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator and a Wi-Fi connection.

3.6.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Launch the ILS application.		Application launched.		Successful	
2	Enter a script in the employee ID field. Enter a valid password.	‘ or 1=1 abc21100217			Successful	
3	Press Login Button.		Employee ID is not valid error message.		Successful	
			Test Case Status	Successful		

3.7 Test Case T0006: Transition from Login to Menu Screen for All roles– Maleeha Masood

3.7.1 Description

After successful login, each user should be shown a menu screen specific to their role.

3.7.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection and valid employee ID and passwords.

3.7.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Launch the ILS application.		Application launched.		Successful	
2	Enter a valid employee ID. Enter a valid password. Press Login button.	21100217 abc21100217	Show Loading Dialog Box and then Manager Menu Screen and clicking on the More button will lead to Manager Specific Menu Screen.		Successful	
3	Repeat steps 1-2 with the different values as in 3.7.4.				Successful	
			Test Case Status	Successful		

3.7.4 *Array of values*

*Note that roles are modifiable and at the time of further testing, employees could have been assigned different roles and so show different screens.

Array of values				
	Scenario 2	**Scenario 3**	**Scenario4**	**Scenario 5**
Value1	21100291	21100326	21100312	21100190
Value2	abc21100291	abc21100326	abc21100312	abc21100190
Expected Result	Show Loading Dialog Box and then Admin Menu Screen and clicking on the More button will lead to Admin Specific Menu Screen.	Show Loading Dialog Box and then Support Menu Screen and clicking on the More button will lead to Support Specific Menu Screen.	Show Loading Dialog Box and then Monitor Menu Screen.	Show Loading Dialog Box and then Initiator Menu Screen.
Actual Result (if different from expected)				
Successful/Failed	Success	Success	Success	Success
Environment Nbr (if failed)				
Log Number (if failed)				

3.8 Test Case T0007: Menu and Specific Menu screens – Aadam Nadeem

3.8.1 Description

A user should be able to view his/her name on the menu screen after logging in and their assigned role on the specific menu screen (if it exists).

3.8.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, user must be logged in with any role.

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Successfully log in to the application with valid credentials	ID: 21100217 Pass: abc21100217	User should be directed to a menu screen with a section title below the banner image Welcoming the user with the text “Welcome Aadam”.		Successful	
2	Go to specific menu screen that exists for Manager, Support and Admin role.		The user should view the screen relevant to his/her role with a title stating the specific role “Manager”.		Successful	
3	Repeat steps 1-2 with logging in from another role and name.		Identical result with specific Name in Main Menu and role in specific menu		Successful	
			Test Case Status	Successful		

3.9 Test Case T0008: Banner Image in Menu and Specific Menu screens – Aadam Nadeem

3.9.1 Description

A user should be able to view a banner image of Allied Bank at the top of the Menu and Specific Menu screen(if it exists).

3.9.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, user must be logged in with any role.

3.9.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Successfully log in to the application with valid credentials	ID: 21100217 Pass: abc21100217	User should be directed to a menu screen with a banner image of Allied bank fetched from a URL displayed at the top		Successful	
2	Go to specific menu screen that exists for Manager, Support and Admin role.		The banner image should be visible beneath the app bar and above the title.		Successful	
			Test Case Status	Successful		

3.9.4 Scenario 2

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Successfully log in to the application with valid credentials	ID: 21100217 Pass: abc21100217	User should be directed to a menu screen with a banner image of Allied bank fetched from a URL displayed at the top	Error in fetching from the image URL	Failed	
2	Go to specific menu screen that exists for Manager, Support and Admin role.		The banner image should be visible beneath the app bar and above the title.	Error in fetching from the image URL	Failed	
			Test Case Status	Failed		

3.10 Test Case T0009: Going back from menu screens – Aadam Nadeem

3.10.1 Description

A user should be directed back to the login screen with the ID and password fields empty.

3.10.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, user must be logged in with any role.

3.3.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Successfully log in to the application with valid credentials	ID: 21100217 Pass: abc21100217	User should be directed to a menu screen with functions according to the relevant role.		Successful	
2	Tap back button from Android phone.		User should be directed back to the login screen with password and ID input fields empty.	The password and ID input fields are filled with the latest entered password and ID.	Failed	
			Test Case Status	Failed		

3.11 Test Case T0010: View info of function in use – Aadam Nadeem

3.11.1 Description

A user will be able to view an info screen containing brief information on the function being accessed and its categories.

3.11.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, user must be logged in with any role.

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	From the Menu/Specific Menu screen, tap on Log an issue button		Directed towards relevant screen		Successful	
2	Tap on the info icon in the right side of the app bar.		User should view a screen with information regarding the overall description of the function functionality and brief descriptions under the headings of its sub-functionalities		Successful	
3	Click on the back arrow at the left of the app bar.		User will be redirected to the previous screen of the function button.		Successful	
4	Repeat steps 1-3 with other buttons depending on functionality		Identical result.		Successful	
			Test Case Status	Successful		

3.12 Test Case T0011: Update Category and Sub-Category lists according to the Area and Category Selection – Malik Ali Hussain

3.12.1 Description

While logging an Issue, the Category and Sub-Category lists should be shown according to the Area and Category selection

3.12.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection and Test Case 0006.

3.12.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Press the Log an Issue button from the Menu		Clicking on the button leads to Log an Issue screen.		Successful	
2	Select an Area. Select a Category.	IT-Support Hardware	Upon selecting these options, the corresponding field gets updated with the selected value and the dropdown lists for subsequent fields get updated with the relevant options.		Successful	
3	Change the Category to Select Category		The Category and Sub-Category dropdown list and values are changed to Select Category and Select Sub-Category.		Successful	
4	Change the Area to Select Area		The Area, Category and Sub-Category dropdown list and values are changed to Select Area, Select Category and		Successful	

			Select Sub-Category.			
5	Repeat steps 1-2 with the different values as in 3.12.4.				Successful	
			Test Case Status	Successful		

	Scenario 2	**Scenario 3**	**Scenario4**
Value1	IT-Support	IT-Support	IT-Support
Value2	Software	Communication	General
Expected Result	Upon selecting these options, the corresponding field gets updated with the selected value and the dropdown lists for subsequent fields get updated with the relevant options.	Upon selecting these options, the corresponding field gets updated with the selected value and the dropdown lists for subsequent fields get updated with the relevant options.	Upon selecting these options, the corresponding field gets updated with the selected value and the dropdown lists for subsequent fields get updated with the relevant options.
Actual Result (if different from expected)			
Successful/Failed	Success	Success	Success
Environment Nbr (if failed)			
Log Number (if failed)			

3.13 Test Case T0012: Log an Issue on behalf of any other employee – Malik Ali Hussain

3.13.1 Description

While logging an Issue, the user should be able to Log an Issue on someone else's behalf.

3.13.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection and Test Case 0006.

3.13.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Press the Log an Issue button from the Menu		Clicking the button leads to Log an Issue screen.		Successful	
2	Press on Log on Behalf checkbox		Clicking the checkbox shows a dialog box for entering the Initiator's details.		Successful	
3	Press Continue without entering any detail		The application does not proceed, instead highlights the empty username and name field as mandatory.		Successful	
4	Enter wrong Initiator Name and Username	Raheem 211003	The user is returned to the Log an Issue screen, a pop-up notification prompts the user that invalid details were entered, and the checkbox is unmarked.		Successful	
5	Press on Log on Behalf checkbox		Clicking the checkbox shows a dialog box for entering the Initiator's details.		Successful	

6	Enter Correct Initiator Name and Username	Raheem Zafar 21100312	The user is returned to the Log an Issue screen and the Log on Behalf checkbox is marked.		Successful	
			Test Case Status	Successful		

3.14 Test Case T0013: Mandatory Fields must be filled to Log an Issue Successfully – Malik Ali Hussain

3.14.1 Description

To Log an Issue successfully, the user must select Branch, Area, Category and Sub-Category and fill the Brief and Detailed Description fields. If any of these fields is not filled, the user must not be able to log an issue and must be prompted to fill the required fields.

3.14.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection and Test Case 0006.

3.14.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Press the Log an Issue button from the Menu		Clicking the button leads to Log an Issue screen.		Successful	
2	Press the Submit button		Clicking the Submit button highlights the empty mandatory fields with red color and the application does not proceed.		Successful	
3	Fill all the mandatory		The Success pop-up is		Successful	

	fields with the details of the issue and then press Submit button		seen, and the user is returned to the Menu screen			
			Test Case Status	Successful		

3.15 Test Case T0014: User must be informed whether Issue was logged successfully or not – Malik Ali Hussain

3.15.1 Description

When a user clicks on Submit button for logging an issue, the user should be informed whether the issue was logged or not. The application may fail to log an issue due to several reasons like no network connectivity, irresponsive server etc. In that case, the user should be informed that the issue has not been logged. If the application is successful in logging the issue then in that case as well, the user should be informed.

3.15.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, and Test Case 0006.

3.15.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Press the Log an Issue button from the Menu		Clicking the button leads to Log an Issue screen.		Successful	
2	Enter the details of the issue, disconnect Wi-Fi and then press the Submit button		Clicking the Submit button shows the Action Failed dialog box and user is returned to Menu screen.		Successful	
3	Reconnect Wi-Fi				Successful	
4	Repeat Step 1				Successful	
5	Enter the details of the issue and press the submit button		Clicking the Submit button shows the Success dialog box and		Successful	

			user is returned to Menu screen.			
			Test Case Status	Successful		

3.16 Test Case T0015: Email is sent to the Initiator and relevant support team upon logging an issue – Malik Ali Hussain

3.16.1 Description

When an issue is logged, email should be sent to the initiator confirming the details of the logged issue and the relevant support team is cc'ed in the email.

3.16.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection and T0006.

3.16.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Press the Log an Issue button from the Menu		Clicking the button leads to Log an Issue screen.		Successful	
2	Enter the details of the issue, and then press the Submit button		Clicking the Submit button shows the Success dialog box and user is returned to Menu screen.		Successful	
3	Check mailbox of the email ID associated with your bank account		An email containing the description of your issue is received and the relevant support team is also cc'ed.		Successful	
			Test Case Status	Successful		

3.17 Test Case T0016: Fetch Lists from DB – Raheem Zafar

3.17.1 Description

All roles to be able to fetch list of issues according to the scenarios listed below.

3.17.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User Logged in as a manager.

3.17.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Click on the 'View your issues' button on the main menu screen.		User should view a list of issues that were logged by the user himself.		Successful	
2	Click on the 'Check all issues' button on the main menu screen.		User should view a list of issues of in-city branches matching their work category.		Successful	
3	Click on the 'Team Issues' button in the 'More'/specific menu screen.		User as a manager should view a list of issues taken up by his team.		Successful	
4	Click on the 'Issues Taken Up' button in the 'More'/ specific menu screen.		User should view a list of issues that were taken up by him/her.		Successful	
			Test Case Status	Successful		

3.18 Test Case T0017: Latest Issue on top – Raheem Zafar

3.18.1 Description

All roles to be able to fetch list of issues according to the scenarios listed below and order of lists is according to the timestamp.

3.18.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User Logged in as a manager.

3.18.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Click on the 'View your issues' button on the main menu screen.		User should view a list of issues that were logged by the user himself. With the issue with the most recent time stamp on top.		Successful	
2	Click on the 'Check all issues' button on the main menu screen.		User should view a list of issues of in-city branches matching their work category. With the issue with the most recent time stamp on top.		Successful	
3	Click on the 'Team Issues' button in the 'More'/ specific menu screen.		User as a manager should view a list of issues taken up by his team. With the issue with the most recent time stamp on top.		Successful	
4	Click on the 'Issues Taken Up' button in the 'More'/ specific menu		User should view a list of issues that were taken up by him/her. With the		Successful	

	screen.		issue with the most recent time stamp on top.			
			Test Case Status	Successful		

3.19 Test Case T0018: Update Status of an Issue (Pop-Up) – Raheem Zafar

3.19.1 Description

A user to be able to view the correct pop-up screen to update the status of an issue logged by him/her.

3.19.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User Logged in as a manager, User to have logged at least one issue before-hand.

3.19.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Click on the ‘View your issues’ button on the main menu screen.		User should view a list of issues that were logged by the user himself.		Successful	
2	Click on any issue in the list.		User should view a popup with the correct details of the issue that was clicked along with two buttons to ‘Mark as Open’ and ‘Mark as Closed’.		Successful	
			Test Case Status	Successful		

3.20 Test Case T0019: Update Status of an Issue – Raheem Zafar

3.20.1 Description

A user to be able to update the status of an issue logged by him/her.

3.20.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User Logged in as a manager, User to have logged at least one issue before-hand.

3.20.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Click on the ‘View your issues’ button on the main menu screen.		User should view a list of issues that were logged by the user himself.		Successful	
2	Click on any issue in the list.		User should view a popup with the correct details of the issue that was clicked along with two buttons to ‘Mark as Open’ and ‘Mark as Closed’.		Successful	
3	Click on “Mark as Open” button.		User should view a ‘Success’ dialog box in case of a successful commit to the DB and for a failed commit the user should view a ‘Action Failed’ dialog box.		Successful	
4	-		The List of issues should be loaded again with		Successful	

			updating the status of the selected Issue			
5	Repeat steps 1-3 with using the 'Mark as Closed button this time'		Results should be same as above.		Successful	
			Test Case Status	Successful		

3.21 Test Case T0020: Email confirmation of issue status update – Raheem Zafar

3.21.1 Description

Users related to the issue status update should receive email notifications accordingly.

3.21.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User Logged in as a manager, User to have logged at least one issue before-hand.

3.21.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/Failed**	**Log Number (if failed)**
1	After committing to update the status of an issue.		Users related to the action (in this case initiator and support staff member) receive email notifications accordingly.		Successful	
			Test Case Status	Successful		

3.22 Test Case T0021: Check all issues - Detail Screen – Raheem Zafar

3.22.1 Description

A user to be able view and take up an issue logged by a person in the same city with the same issue relevance area.

3.22.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User Logged in as a manager, there must be an issue logged in the same city and with the same relevant area.

3.22.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Click on the 'Check all Issues' button on the main menu screen.		User should view a list of issues that were logged in the same city as the user as well as have the same area as the user.		Successful	
2	Click on any issue in the list.		User should view a screen with the correct details of the issue that was clicked along with a "Take up" button.		Successful	
			Test Case Status	Successful		

3.23 Test Case T0022: Take up an Issue – Raheem Zafar

3.23.1 Description

A user to be able view and take up an issue logged by a person in the same city with the same issue relevance area.

3.23.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User Logged in as a manager, there must be an issue logged in the same city and with the same relevant area.

3.23.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Click on the 'Check all issues.' button on the main menu screen.		User should view a list of issues that were logged by the user himself.		Successful	
2	Click on any issue in the list.		User should view a screen with the correct details of the issue that was clicked along with a "Take up" button.		Successful	
3	Click on the "Take Up" button.		User should view a 'Success' dialog box in case of a successful commit to the DB and for a failed commit the user should view a 'Action Failed' dialog box.		Successful	
5	-		The Issue status icon should change to the blue icon in the check all issues List and the DB should assign the issue to the user.		Successful	

Test Case Status	Successful

3.24 Test Case T0023: Email Confirmation Issue taken up – Raheem Zafar

3.24.1 Description

A user to be able to update the status of an issue logged by him/her and receive notifications accordingly.

3.24.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User Logged in as a manager, User to have logged at least one issue before-hand.

3.24.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/Failed**	**Log Number (if failed)**
1	After committing to take up an issue.		Users related to the action (in this case initiator and support staff member who took up the issue) receive email notifications accordingly.		Successful	

Test Case Status	Successful

3.25 Test Case T0024: Issues taken up - Detail Screen – Raheem Zafar

3.25.1 Description

A user to be able view the detail of an issue already taken up by him/her.

3.25.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User must have taken up an issue.

3.25.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Click on the 'Issues Taken Up' button in the More/ specific menu screen.		User should view a list of issues that were taken up by him/her.		Successful	
2	Click on any issue in the list.		User should view a User should view a screen with the correct details of the issue that was clicked along with a dropdown menu "actions" and an "update" button.		Successful	
			Test Case Status	Successful		

3.26 Test Case T0025: Issues taken up- Action Commit (Mar as open and Mark as Fixed) - Detail Screen – Raheem Zafar

3.26.1 Description

A user will be able to change the status of an issue he/she has taken up.

3.26.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User must have taken up an issue.

3.26.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Click on the 'Issues Taken Up' button in the More/ specific menu screen.		User should view a list of issues that were taken up by him/her.		Successful	
2	Click on any issue in the list.		User should view a screen with the correct details of the issue that was clicked along with a dropdown menu "actions" and an update button.		Successful	
3	Click on the action dropdown menu.		User should be able to view a menu that contains three options. "Mark as fixed, Mark as Open and Re-Assign"		Successful	
4	Click on the Mark as Fixed option.		A success dialog should be viewed, and the status of the Issue should change to "fixed" in the database and the list when loaded again. For a failed commit, an "Action Failed" dialog box should be viewed.		Successful	
5	Repeat steps 1-4 with the Mark as open option instead.		A success dialog should be viewed, and the status of the Issue should change to "open" in the database and the list when loaded again.		Successful	

			For a failed commit, an "Action Failed" dialog box should be viewed.			
			Test Case Status	Successful		

3.27 Test Case T0026: List Employees to re-assign issues to. – Aadam Nadeem

3.27.1 Description

A user will be able to view a list of employees from the relevant support team to reassign the issue selected to.

3.27.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, user must be logged in with manager credentials and that there must exist at least one issue in the system.

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/Failed**	**Log Number (if failed)**
1	Click on the 'Issues Taken Up' button in the More/ specific menu screen.		User should view a list of issues that were taken up by him/her.		Successful	
2	Click on any issue in the list.		User should view a screen with the correct details of the issue that was clicked along with a		Successful	

			dropdown menu “actions” and an update button.			
3	Click on the action dropdown menu.		User should be able to view a menu that contains three options. “Mark as fixed, Mark as Open and Re-Assign”		Successful	
4	Select the ‘Reassign’ option. Tap on the Update button		The user should be directed to a new screen with the list of details (Name and Role) of the relevant support team employees.		Successful	
			Test Case Status	Successful		

3.28 Test Case T0027: Search for employees to re-assign issues to. – Aadam Nadeem

3.28.1 *Description*

A user will be able to search for an employee in the reassign screen to reassign the issue selected to.

3.28.2 *Pre-conditions for this test case*

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, user must be logged in with manager credentials and that there must exist at least one issue in the system.

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/Failed**	**Log Number (if failed)**
1	Click on the 'Issues Taken Up' button in the More/ specific menu screen.		User should view a list of issues that were taken up by him/her.		Successful	
2	Click on any issue in the list.		User should view a screen with the correct details of the issue that was clicked along with a dropdown menu "actions" and an update button.		Successful	
3	Click on the action dropdown menu.		User should be able to view a menu that contains three options. "Mark as fixed, Mark as Open and Re-Assign"		Successful	
4	Select the 'Reassign' option. Tap on the Update button		The user should be directed to a new screen with the list of details (Name and Role) of the relevant support team employees with the option to search for a specific employee in the database.		Successful	
5	Enter a valid employee name. Enter a valid employee	Aadam Nadeem 21100190	The employee will appear and can be reassigned the issue to.		Successful	

	ID. Press Search button					
6	Repeat step 1 – 5 with invalid credentials	Aadam Nadeem 21100	A dialog box should appear with the message that user does not exist	Blank screen with no employee details tile	Failed	0004
			Test Case Status	Failed		

3.29 Test Case T0028: SQL injections in Search for employees to re-assign issues to. – Aadam Nadeem

3.29.1 Description

In the search categories, users are prompted to enter in valid employee names and IDs. To avoid a scenario where attacker can cause an XSS attack by using script tags, a user should not be allowed to enter scripts in the employee ID field.

3.29.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, user must be logged in with manager credentials and that there must exist at least one issue in the system.

3.29.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/Failed**	**Log Number (if failed)**
1	Launch the ILS application.		Application launched.		Successful	
2	Click on any issue in the list.		User should view a screen with the correct details of the issue that was clicked along with a		Successful	

			dropdown menu "actions" and an update button.			
3	Click on the action dropdown menu.		User should be able to view a menu that contains three options. "Mark as fixed, Mark as Open and Re-Assign"		Successful	
4	Select the 'Reassign' option. Tap on the Update button		The user should be directed to a new screen with the list of details (Name and Role) of the relevant support team employees with the option to search (in lower right corner) for a specific employee in the database.		Successful	
5	Enter a name. Enter a script in the employee ID field.	Aadam Nadeem <script>alert('HI ')</script>			Successful	
6	Press Search button		Employee ID is not valid error message	Proceeds to run script	Failed	
			Test Case Status	Failed		

3.30 Test Case T0029: Reassign an issue to an employee. – Aadam Nadeem

3.30.1 Description

A user will be able to successfully reassign an issue to a selected member of the support team.

3.30.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, user must be logged in with manager credentials and that there must exist at least one issue in the system.

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/Failed**	**Log Number (if failed)**
1	Click on the 'Issues Taken Up' button in the More/ specific menu screen.		User should view a list of issues that were taken up by him/her.		Successful	
2	Click on any issue in the list.		User should view a screen with the correct details of the issue that was clicked along with a dropdown menu "actions" and an update button.		Successful	
3	Click on the action dropdown menu.		User should be able to view a menu that contains three options. "Mark as fixed, Mark as Open and Re-Assign"		Successful	
4	Select the 'Reassign'		The user should be		Successful	

	option. Tap on the Update button		directed to a new screen with the list of details (Name and Role) of the relevant support team employees. Upon selection, an employee's details can be seen on a new screen with an assign button at the end.			
5	Select an employee and tap on the 'assign' button		Dialog box will appear stating that the issue has been successfully reassigned.		Successful	
6	Repeat steps 1-5 with the search option as stated in 3.24		Dialog box will appear stating that the issue has been successfully reassigned.		Successful	
			Test Case Status	Successful		

3.31 Test Case T0030: Email Confirmation upon re-assigning an issue – Aadam Nadeem

3.31.1 Description

A user to be able to reassign an issue to another employee of the support team and receive notifications accordingly.

3.31.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, user logged in as a manager, user to have logged at least one issue in the system before-hand.

3.31.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/Failed**	**Log Number (if failed)**
1	Reassigning an issue as explained in test case 3.26		Users related to the action (in this case manager and support team employee to whom the issue has been reassigned) receive email notifications accordingly.		Successful	
			Test Case Status	Successful		

3.32 Test Case T0031: User must be informed whether Issue was reassigned successfully or not– Aadam Nadeem

3.32.1 Description

The user should view the appropriate dialog box while assigning an issue to the user.

3.32.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, user logged in as a manager, user to have logged at least one issue in the system before-hand.

3.32.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/Failed**	**Log Number (if failed)**
1	Reassigning an issue as explained in test case 3.26		If data has been successfully fetched and/or sent to the database, a "Success!" dialog box will appear		Successful	
2	Repeat step 1		If there is an issue with committing to the database, an "Action Failed!" dialog box should appear after tapping 'Assign'		Successful	
			Test Case Status	Successful		

3.33 Test Case T0032: Manager Team Issues - Detail Screen – Shahrukh Kemall

3.33.1 Description

A manager will be able to view the details of an issue taken up by a member of his support team.

3.33.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User must be a manager and his team must have taken up at least one issue.

3.33.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	In the More menu for a manager click on 'Team Issues'.		User should view a list of issues that were taken up by members of his/her support staff team.		Successful	
2	Click on any issue in the list.		User should view a screen with the correct details of the issue that was clicked along with a "re-assign" button.		Successful	
			Test Case Status	Successful		

3.34 Test Case T0033: Reassign Issue- Button – Shahrukh Kemall

3.34.1 Description

A manager will be able to view the list of employees an issue can be re-assigned to.

3.34.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User must be a manager, his team must have taken up at least one issue and there should be at least one other employee in the DB.

3.34.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	In the More menu for a manager click on 'Team Issues'.		User should view a list of issues that were taken up by members of his/her support staff team.		Successful	
2	Click on any issue in the list.		User should view a screen with the correct details of the issue that was clicked along with a "re-assign" button.		Successful	
3	Click on the re-assign button.		User should be able to view a screen that shows a list of employees that the issue can be re-assigned along with a search button.		Successful	
			Test Case Status	Successful		

3.35 Test Case T0034: Reassign Issue - Employee detail screen – Shahrukh Kemall

3.35.1 Description

A manager will be able to view the details of an employee he/she wishes to assign an issue.

3.35.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User must be a manager, his team must have taken up at least one issue and there must be at least one more employee in the DB

3.35.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	In the More menu for a manager click on 'Team Issues'.		User should view a list of issues that were taken up by members of his/her support staff team.		Successful	
2	Click on any issue in the list.		User should view a screen with the correct details of the issue that was clicked along with a "re-assign" button.		Successful	
3	Click on the re-assign button.		User should be able to view a screen that shows a list of employees that the issue can be re-assigned along with a search button.		Successful	
4	Click on any employee entry on the screen		User should be able to view a screen showing correct employee details of the employee that was clicked on along with a		Successful	

			‘assign’ button			
			Test Case Status	Successful		

3.36 Test Case T0035: Reassign Issue – By List – Shahrukh Kemall

3.36.1 Description

A manager will be able to re-assign an issue to any employee from the list shown.

3.36.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User must be a manager, his team must have taken up at least one issue and there must be at least one more employee in the DB

3.36.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	In the More menu for a manager click on ‘Team Issues’.		User should view a list of issues that were taken up by members of his/her support staff team.		Successful	
2	Click on any issue in the list.		User should view a screen with the correct details of the issue that was clicked along with a “re-assign” button.		Successful	
3	Click on the re-assign button.		User should be able to view a screen that shows a list of employees that the issue can be re-assigned along with a search button.		Successful	

4	Click on any employee entry on the screen		User should be able to view a screen showing correct employee details of the employee that was clicked on along with a ‘assign’ button		Successful	
5	Click on the “assign” button.		In case of a successful commit to the DB the application should show a ‘success’ dialog box and in case of a failed commit an ‘Action Failed’ dialog should be shown.			
			Test Case Status	Successful		

3.37 Test Case T0036: Reassign Issue – Search PopUp and Employee Detail Screen – Shahrukh Kemall

3.37.1 Description

A manager will be able to re-assign an issue to any employee when searched from the DB.

3.37.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User must be a manager, his team must have taken up at least one issue and there must be at least one more employee in the DB

3.37.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	In the More menu for a manager click on 'Team Issues'.		User should view a list of issues that were taken up by members of his/her support staff team.		Successful	
2	Click on any issue in the list.		User should view a screen with the correct details of the issue that was clicked along with a "re-assign" button.		Successful	
3	Click on the re-assign button.		User should be able to view a screen that shows a list of employees that the issue can be re-assigned along with a search button.		Successful	
4	Click on the search button.		A PopUp should show with fields for Employee Name and Username.		Success	
5	Enter Correct Employee details.	Raheem Zafar 21100312	A screen should show with the correct details of the employee whose details were entered.	A screen is shown with the correct format, but the employee detail fields are blank.	Failed	
	Enter wrong employee details.	RZafar 211	A dialog should appear that says no such user exists.	A screen is shown with the correct format, but the employee detail fields are blank.	Failed	

Test Case Status	Failed

3.38 Test Case T0037: Email Confirmation Issue re-assigned – Shahrukh Kemall

3.38.1 Description

A manager to be able to re-assign and issue and receive notifications accordingly.

3.38.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, User Logged in as a manager, User to have logged at least one issue before-hand.

3.38.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/Failed**	**Log Number (if failed)**
1	After committing to re-assign an issue.		Users related to the action (in this case initiator and support staff member who took up the issue) receive email notifications accordingly.		Successful	
			Test Case Status	Successful		

3.39 Test Case T0038: Employee list is displayed to Admin when changing an employee’s attribute – Malik Ali Hussain

3.39.1 Description

To change the role or the team of an employee, Admin should be able to select him/her from the list of employees shown on the screen.

3.39.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection and Test Case 0006.

3.39.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Press the Change Team button from the Menu.		Clicking the button leads to Change Team screen which shows a list of names and roles of the employees currently in the database.		Successful	
2	Press the back arrow from the app bar		The user is returned to the Menu screen		Successful	
3	Press the Change Role button from the Specific Menu Screen that appears after pressing More button on the Menu Screen.		Clicking the button leads to Change Role screen which shows a list of names and roles of the employees currently in the database.		Successful	
			Test Case Status	Successful		

3.40 Test Case T0039: Directly search an employee from the database for changing the attribute – Malik Ali Hussain

3.40.1 Description

For changing an attribute of an employee, the Admin should be able to search him/her directly from the database.

3.40.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection and Test Case 0000.

3.40.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Press Search button on the bottom right corner of the screen		Clicking the button shows a dialog box for entering the employee's details.		Successful	
2	Press Continue without entering any detail		The application does not proceed, instead highlights the empty username and name field as mandatory.		Successful	
3	Enter wrong Name and Username	Raheem 211003	The user is returned to the change attribute screen, a pop-up notification prompts the user that invalid details were entered.		Successful	
4	Press Search button on the bottom right corner of the screen		Clicking the button shows a dialog box for entering the employee's details.		Successful	
5	Enter Correct Name and Username	Raheem Zafar 21100312	The user is navigated to the screen which shows all the details of that		Successful	

			employee and has the option to change the attribute.			
			Test Case Status	Successful		

3.41 Test Case T0040: Admin is able to see the details of the employee before changing the employee's attribute – Malik Ali Hussain

3.41.1 Description

Before changing the role or the team of an employee, Admin should be able to see the details of the selected employee. These details include the employee's name, username, role, team and expertise (in case the Admin is changing the team).

3.41.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection and Test Case 0000.

3.41.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Select the relevant employee from the list	Raheem Zafar	The Admin is navigated to the screen which shows all the details of that employee mentioned in 3.19.1 and has the option to change the attribute.		Successful	
			Test Case Status	Successful		

3.42 Test Case T0041: New Attribute must be selected before updating the employee details – Malik Ali Hussain

3.42.1 Description

To Change the Role or Team successfully, the Admin must select Team or Branch (according to the screen). If this fields is not filled, the user must not be able to update the details of the employee and must be prompted to fill the required field.

3.42.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection and Test Case 0000.

3.42.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Press the Update button		Clicking the Update button highlights the empty mandatory field with red color and the application does not proceed.		Successful	
2	Select an option from the dropdown list in the change attribute field and press Update button		The Success pop-up is seen, and the user is returned to the screen with the employee list		Successful	
			Test Case Status	Successful		

3.43 Test Case T0042: Admin must be informed whether the attribute was changed successfully or not – Malik Ali Hussain

3.43.1 Description

When the Admin clicks on Update button for changing the attribute of an employee, the user should be informed whether the issue was logged or not. The application may fail to update the database with the change due to several reasons like no network connectivity, irresponsive server etc. In that case, the user should be informed that the attribute was not changed. If the application is successful then in that case as well, the user should be informed.

3.43.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection and Test Case 0000.

3.43.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Log Number (if failed)**
1	Select an option from the dropdown list in the change attribute field, disconnect the Wi-Fi and press Update button		Clicking the Update button shows the Action Failed dialog box and user is returned to the screen with the employee list		Successful	
2	Reconnect Wi-Fi				Successful	
3	Select an option from the dropdown list in the change attribute field and press Update button		The Success pop-up is seen, and the user is returned to the screen with the employee list		Successful	
			Test Case Status	Successful		

3.44 Test Case T00043: Generate Report for Manager and Monitor Roles– Maleeha Masood

3.44.1 Description

After successful login, Manager and Monitor role users should be shown a button to Generate Reports.

3.44.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection and valid employee ID and passwords.

3.44.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/Failed**	**Log Number (if failed)**
1	Launch the ILS application.		Application launched.		Successful	
2	Enter a valid employee ID. Enter a valid password. Press Login button.	21100217 abc21100217	Show Loading Dialog Box and then Manager Menu Screen.		Successful	
3	Click on More button on Menu.		Specific Menu Screen appears.			
4	Click on Generate Report button.		Screen with graphs appear. Swipe to see more graphs.			
5	Repeat steps 1, 2, 4 with the different values as in 3.9.4.				Successful	
			Test Case Status	Successful		

3.44.4 *Array of values*

*Note that roles are modifiable and at the time of further testing, employees could have been assigned different roles and so show different screens.

Array of values	
	Scenario 2
Value1	21100312
Value2	abc21100312
Expected Result	Screen with graphs appear. Swipe to see more graphs.
Actual Result (if different from expected)	
Successful/Failed	Success
Environment Nbr (if failed)	
Log Number (if failed)	

3.45 Test Case T0044: Graphs Fetching Data Real-Time as Database Gets Updated – Maleeha Masood

3.45.1 *Description*

Manager and Monitor role users should be able to view changed graphs as the database gets updated.

3.45.2 *Pre-conditions for this test case*

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, Test Case 0008, 2 users and Test Case (FOR LOG AN ISSUE).

3.45.3 *Scenario 1*

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/Failed**	**Log Number (if failed)**
1	Once on the graph					

	screen, user 1 should wait and observe.					
2	On another device, user 2 should login and log 10 new issues.		User 1 should view the graphs changing.	User 1 does not view any change in the graph.	Failed	003
			Test Case Status	Failed		

3.46 Test Case T0045: Back Button of the Multiple Graphs of Generate Report Screen – Maleeha Masood

3.46.1 Description

Manager and Monitor role users should be able to go back to the menu screen from the Generate Report Screen.

3.46.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection, and Test Case 0008.

3.46.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/Failed**	**Log Number (if failed)**
1	After viewing the Generate Report Screen, press the back button from the first graph.		User should view the specific menu or the menu screen again depending on their role.		Successful	
2	Repeat step 1 two times, once from the second graph (viewed by swiping to the left once) and once from the third graph (viewed by swiping to the left twice).		User should view the specific menu or the menu screen again depending on their role.			

Test Case Status	Successful

3.47 Test Case T0046: Logout from Any Role– Maleeha Masood

3.47.1 Description

All users should be able to logout from their accounts.

3.47.2 Pre-conditions for this test case

Installed apk of the application either on an Android phone or an Android emulator, a Wi-Fi connection and Test Case 0002.

3.47.3 Scenario 1

Test Case						
UC Step	**Step Description**	**Data/Value (1 to n)**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/Failed**	**Log Number (if failed)**
1	Depending on the role, find the logout button either on the Menu Screen or on the Specific Menu Screen that appears after pressing More button on the Menu Screen.					
2	Press the Logout Button.		Login Screen Appears.			

Test Case Status	Successful

Summary of manual testing results – All Team Members

Overall, the testing results of the main use cases of our application were mostly positive and were handled correctly. The core functionalities of the application work correctly on all edge cases. However, there exist some minor errors in the secondary functions. Few things which could have been improved are the responsiveness of the user interface to the actions and responsiveness of the backend server, some of which include delays in fetching certain data from the backend database. These micro issues could be handled by a loading screen at some points where the data is taking longer than expected to load, like we handled in the log in screen. Security threats like Sql and XSS attacks were catered for in placeholders in the login screen, while only half implemented while searching for an employee within the app, which is a secondary functionality. The overall mainframe of the application is responsive to all user actions whether it is by a dialog message or a slight change in color on pressing the button. Moreover, extra information of the employees is kept safe as users can only view/search, issues/employees available in the current area of the branch.

Aadam Nadeem

A total of ten test cases were checked of which 4 test cases failed and 6 of them were successful. Of the four test cases that failed, two were a security risk of which one was minor while another one about not erasing ID and password was a major security risk not catered. Another one of the issues was committing to the database in one module (Reassign issue). However, that case is catered by displaying the appropriate dialog box to the user in case of a success or a failure. A slight delay with fetching lists from the database can cause the user to think maybe it is empty without a loading dialog box to assure them that the application is still fetching data. Aside from that, all edge cases of the use cases I wrote about were working successfully and if not, were successfully catered to.

Maleeha Masood

Out of the ten tests that I ran, 3 tests failed. 2 by 3 of the tests that failed were on the micro level details that were not directly emphasized on by the functional and the non-functional requirements. Both these tests were on the extra tiny details that might have made our application look more mature to the user. The remaining test that failed was due to issues in the fetching of data from database. While not entirely visible from my tests, fetching data from the database was a prominent issue that I could feel while performing tests on the application. The lag of data retrieval was too high and there was barely any synchronization between the UI and data fetching. I believe this is something that the ILS app needs more.

Malik Ali Hussain

I performed 10 tests manually on the application and 9 out of 10 passed. The test which failed was to search employee directly from database. It was working in the development phase but probably due to some error the code got messed up, which can be fixed in the following update and provided to you for evaluation. This shows that the main functionality of the use cases I tested has been implemented. Although all of the tests passed but I still found some shortcomings in our applications which needs improvement. Firstly, while changing the attribute of an employee if we click on the Change Role or Change Team button, a list is shown for selecting the relevant person. While the application is fetching the data from the database, the table shows 'Loading' in the name and role fields. If we click on any option before the table is updated, then the application proceeds to the next screen but does not the details of any person. Moreover, if the internet is disconnected at any point and an action is performed like log an issue or update the attribute etc., the action fails, and the dialog box shown to the user only states 'Action Failed'. It does not tell the actual reason for the failure, while it fulfills the basic requirement of prompting the user, in case an action fails, but in my opinion the alert can be made more specific to let the user know the actual cause of failure as well. These all shortcomings can be fulfilled in an update for the application which can be made once we get the prototype or the initial version of our application approved by our client.

Raheem Zafar

I chose the test cases accordingly to avoid redundancy and to cover up maximum number amount of functionality to be tested. All test cases tested are a part of the core functionality of the application. Talking about the flaws, all test cases I tested lacked a feature that the user was never notified when internet access was not available. I did not include these in the test cases as including it on all steps would have made it redundant. Another flaw that was recognized was that is the Database has no data for a specific list the application should show a message which tells that there are no issues preset in this category but instead it shows a blank field. Apart from these all the tests were carried out thoroughly, for testing purposes we had limited records in lists in our DB and the lists viewed on screen were manually tallied with the data present in the DB through launching manual queries on MySql workbench. Other components like committing data writes or updates to the DB could be tested through the apk itself and were double checked by confirming the API responses.

4. Automated testing

The purpose of Automated testing is to verify accurate functionality of the API calls sent from the services.dart’s AppServices class functions. This assures the validity of API calls and data returned from the API called without actually using the application features.

The scope of this automated testing is limited to all API calls made by the applications. Action from each function within services.dart’s AppServices class, correspond to an API call that reads or updates the data from the database.

This section was a team effort by all 5 members of the team.

Maleeha Masood worked on flutter testing, details of which are in section 5 below. Malik, Raheem, Shahrukh and Aadam collectively worked on API testing in this section where each member chose an API testing tool to apply and move forward with. These included SOAP UI, Postman, Dreamfactory and JMeter. After thorough research and application, the team decided to use x as an API testing tool.

4.1 List of test cases used for automated testing

Testcase ID	Testcase name	Performed by	Successful/Failed
API-001	Validating Services.dart API call with action ‘LOG_ISSUE’, should log an issue in Issues table	Shahrukh Kemall	Successful

API-002	Validating Services.dart API call with action ‘VALIDATE_EMP’, it validates Employee ID and Employee Password while login	Shahrukh Kemall	Successful
API-003	Validating Services.dart API call with action ‘VALIDATE_EMP_ID’, it checks if an employee ID exists in the database	Shahrukh Kemall	Successful
API-004	Validating Services.dart API call with action ‘_GET_ALL_ISSUES_BY_EMP’, it returns issues logged by the Employee ID	Shahrukh Kemall	Successful
API-005	Validating Services.dart API call with action ‘_GET_ALL_ISSUES’, it returns all issues from the issues table	Shahrukh Kemall	Successful
API-006	Validating Services.dart API call with action ‘UPDATE_STATUS’, it updates Current_Status of the issue logged in issues table	Shahrukh Kemall	Successful

API-007	Validating Services.dart API call with action ‘_GET_ALL_LOCATIONS’, it returns list of all branches details from the branch table	Shahrukh Kemall	Successful
API-008	Validating Services.dart API call with action ‘_GET_ALL_BRANCH’, it returns list of all Branch_IDs from the branch table	Shahrukh Kemall	Successful
API-009	Validating Services.dart API call with action ‘_GET_ALL_AREA’, it returns list of all unique Area of Expertise from the support_team table	Shahrukh Kemall	Successful
API-010	Validating Services.dart API call with action ‘_GET_ALL_CATEGORY’, it returns list of all unique categories of support_team member expertise from the support_team table	Shahrukh Kemall	Successful

API-011	Validating Services.dart API call with action '_GET_ALL_SUBCATEGORY', it returns list of all unique sub-categories of support_team member expertise and category from the support_team table	Shahrukh Kemall	Successful
API-012	Validating Services.dart API call with action 'GET_NAME', it returns name of the employee against an Employee ID	Shahrukh Kemall	Successful
API-013	Validating Services.dart API call with action 'GET_ROLE', it returns role of the employee against an Employee ID	Shahrukh Kemall	Successful
API-014	Validating Services.dart API call with action '_GET_ISSUE_BY_ISSUE_ID', it returns details of the issue against its Issue_ID	Shahrukh Kemall	Successful

API-015	Validating Services.dart API call with action '_GET_ALL_LOCATIONS_BY_ID', it returns branch details against a Branch_ID from the branch table	Shahrukh Kemall	Successful
API-016	Validating Services.dart API call with action 'VALIDATE_NAME_OF_EMP', it validates an employee ID against the name of the employee in the employee table	Malik Ali Hussain	Successful
API-017	Validating Services.dart API call with action '_GET_ALL_ISSUES_BY_MANAGERS_TEAM', it returns issues taken up by a manager's team	Malik Ali Hussain	Successful
API-018	Validating Services.dart API call with action '_GET_YOUR_TAKEN_ISSUES', it returns issues taken up by a Support Team member	Malik Ali Hussain	Successful

API-019	Validating Services.dart API call with action '_GET_ALL_INCITY_ISSUES', it returns all relevant issues with respect to support_team member's location	Malik Ali Hussain	Successful
API-020	Validating Services.dart API call with action 'UPDATE_ROLE', it updates role of the employee against an Employee ID	Malik Ali Hussain	Successful
API-021	Validating Services.dart API call with action 'UPDATE_TEAM', it changes team of the employee against an Employee ID	Raheem Zafar	Successful
API-022	Validating Services.dart API call with action 'ASSIGN_ISSUE', it assigns an issue to an Employee ID by updating issue_taken_up in issues table	Raheem Zafar	Successful

API-023	Validating Services.dart API call with action '_GET_EXPERTISE', it returns details (Name, Employee_ID, Role) of all employees	Raheem Zafar	Successful
API-024	Validating Services.dart API call with action '_GET_EXPERTISE_BY_ID', it returns details (Name, Employee_ID, Role) of an Employee	Raheem Zafar	Successful
API-025	Validating Services.dart API call with action '_GET_Category_BY_ID', it returns all distinct categories in an area of expertise of a Support Team Member	Aadam Nadeem	Successful
API-026	Validating Services.dart API call with action 'COUNT_FROM_ISSUES', it returns number of issues logged in issues table where the 'key' column contains the 'val' value, both provided	Aadam Nadeem	Successful

API-027	Validating Services.dart API call with action '_GET_EMAIL', it returns email stored against an Employee ID in employee table	Aadam Nadeem	Successful
API-028	Validating Services.dart API call with action '_GET_Team_EMAILS', it returns list of emails of relevant support team members when issue within their team/Branch ID and category of their area of expertise is logged	Aadam Nadeem	Successful
API-029	Validating Services.dart API call with action '_GET_CITY_EMAILS', it returns list of emails of relevant support team members when issue within their city and category of their area of expertise is logged	Aadam Nadeem	Successful

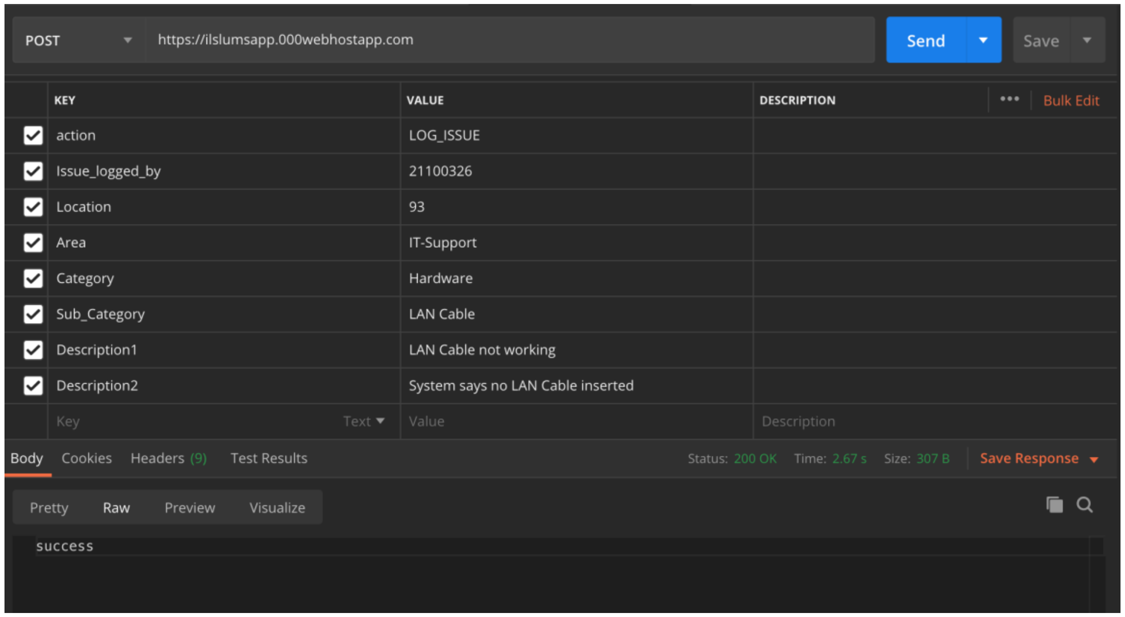
4.2 Postman

Postman is the application used for testing APIs. It is used to send http request to the web server and getting the response back. This tool allows you check if your API meets expectations for functionality, reliability and performance and returns the correct response. It is a very user friendly software .

4.3 Setup

Postman can either be accessed via its website or the application can be downloaded for frequent use. The application is available for all Windows/MacOS/Linux environments. The signup is free of cost unlike SoapUI that only gives a free trail. However, a working internet connection is required to send Http requests.

4.4 Test cases for automated testing



https://ilslumsapp.000webhostapp.com

KEY	VALUE	DESCRIPTION
action	LOG_ISSUE	
Issue_logged_by	21100326	
Location	93	
Area	IT-Support	
Category	Hardware	
Sub_Category	LAN Cable	
Description1	LAN Cable not working	
Description2	System says no LAN Cable inserted	

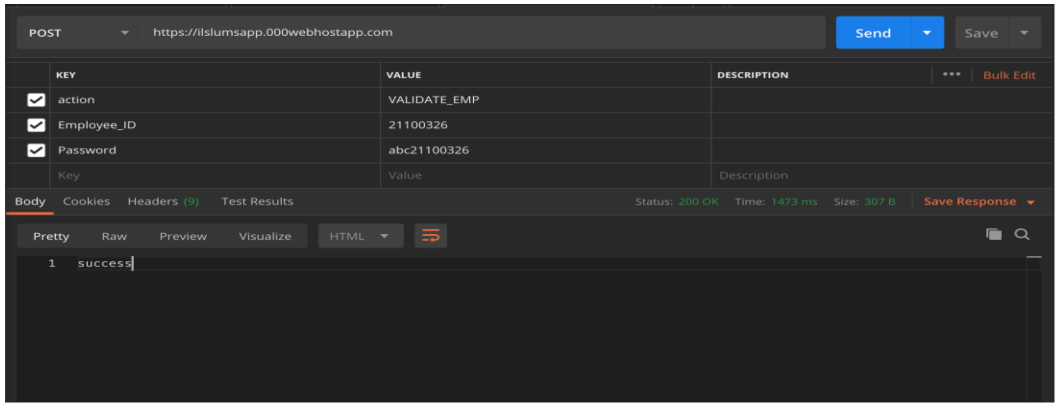
Status: 200 OK Time: 2.67 s Size: 307 B

```
success
```

4.4.1 Testcase ID: API-001

Description: Validating Services.dart API call with action ‘LOG_ISSUE’, should log an issue in Issues table. Returns success.

Result: API call is valid.

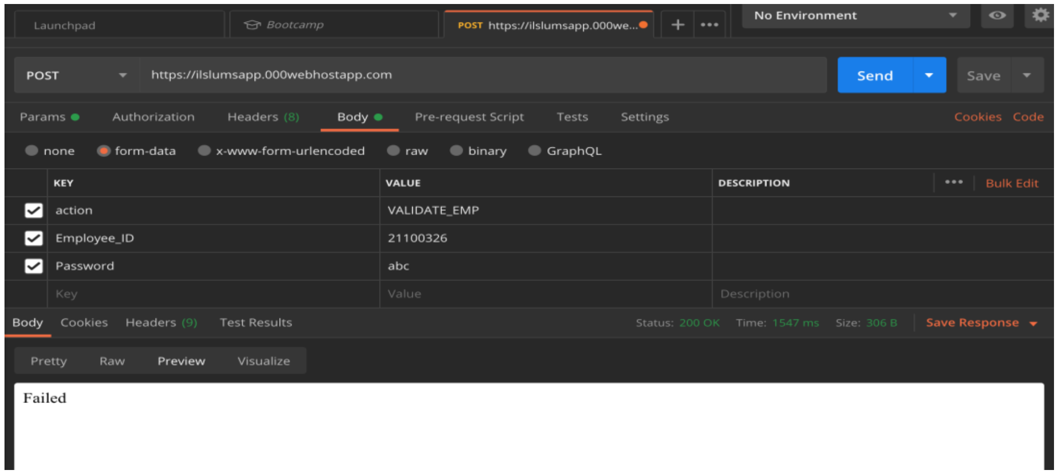


POST https://ilslumsapp.000webhostapp.com

KEY	VALUE	DESCRIPTION
action	VALIDATE_EMP	
Employee_ID	21100326	
Password	abc21100326	

Status: 200 OK Time: 1473 ms Size: 307 B

```
success
```



POST https://ilslumsapp.000webhostapp.com

Body: form-data

KEY	VALUE	DESCRIPTION
action	VALIDATE_EMP	
Employee_ID	21100326	
Password	abc	

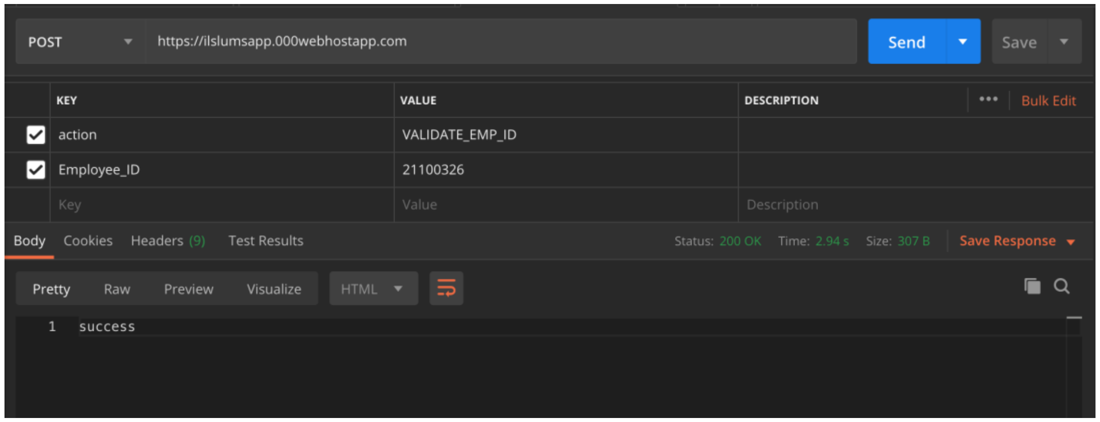
Status: 200 OK Time: 1547 ms Size: 306 B

```
Failed
```

4.4.2 Testcase ID: API-002

Description: Validating Services.dart API call with action ‘VALIDATE_EMP’, it validates Employee ID and Employee Password while login. Return success on valid Employee_ID and Password and failed when either is incorrect

Result: API call is valid

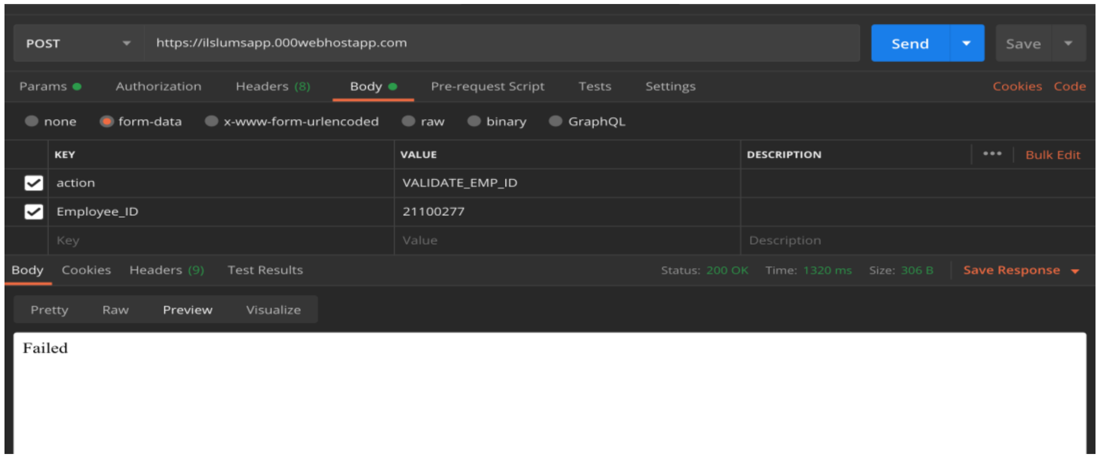


POST https://ilslumsapp.000webhostapp.com

KEY	VALUE	DESCRIPTION
action	VALIDATE_EMP_ID	
Employee_ID	21100326	

Status: 200 OK Time: 2.94 s Size: 307 B

```
success
```



POST https://ilslumsapp.000webhostapp.com

Body: form-data

KEY	VALUE	DESCRIPTION
action	VALIDATE_EMP_ID	
Employee_ID	21100277	

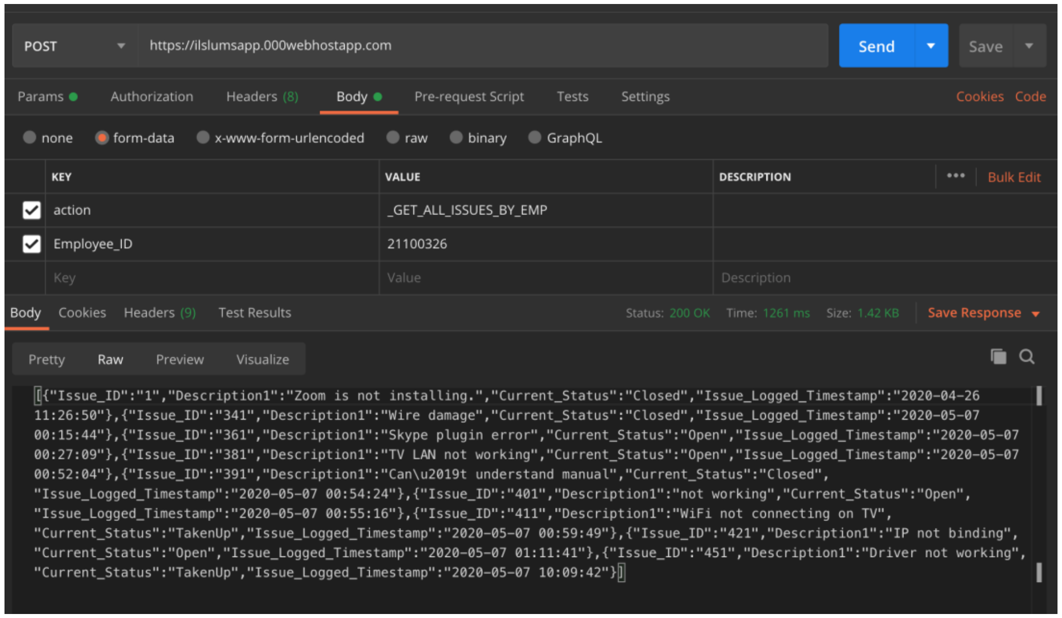
Status: 200 OK Time: 1320 ms Size: 306 B

```
Failed
```

4.4.3 Testcase ID: API-003

Description: Validating Services.dart API call with action ‘VALIDATE_EMP_ID’, it checks if an employee ID exists in the database. Returns success if valid. Returns failed is invalid.

Result: API call is valid.



KEY	VALUE	DESCRIPTION
action	_GET_ALL_ISSUES_BY_EMP	
Employee_ID	21100326	

Status: 200 OK Time: 1261 ms Size: 1.42 KB

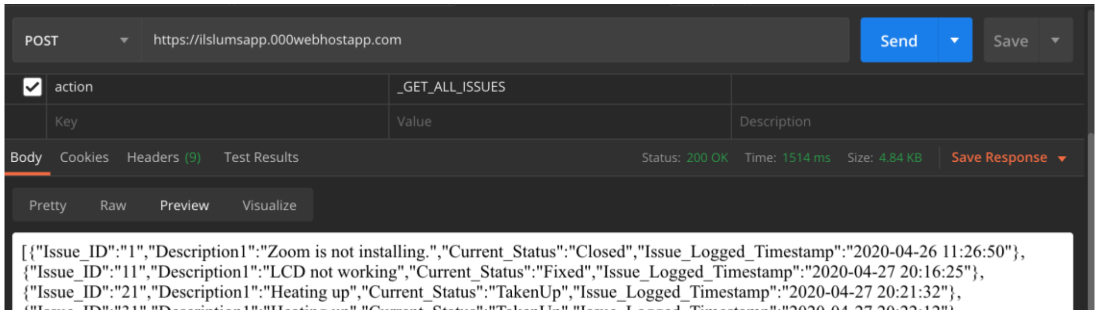
```
[{"Issue_ID":"1","Description1":"Zoom is not installing.","Current_Status":"Closed","Issue_Logged_Timestamp":"2020-04-26
11:26:50"},{"Issue_ID":"341","Description1":"Wire damage","Current_Status":"Closed","Issue_Logged_Timestamp":"2020-05-07
00:15:44"},{"Issue_ID":"361","Description1":"Skype plugin error","Current_Status":"Open","Issue_Logged_Timestamp":"2020-05-07
00:27:09"},{"Issue_ID":"381","Description1":"TV LAN not working","Current_Status":"Open","Issue_Logged_Timestamp":"2020-05-07
00:52:04"},{"Issue_ID":"391","Description1":"Can\u2019t understand manual","Current_Status":"Closed",
"Issue_Logged_Timestamp":"2020-05-07 00:54:24"},{"Issue_ID":"401","Description1":"not working","Current_Status":"Open",
"Issue_Logged_Timestamp":"2020-05-07 00:55:16"},{"Issue_ID":"411","Description1":"WiFi not connecting on TV",
"Current_Status":"TakenUp","Issue_Logged_Timestamp":"2020-05-07 00:59:49"},{"Issue_ID":"421","Description1":"IP not binding",
"Current_Status":"Open","Issue_Logged_Timestamp":"2020-05-07 01:11:41"},{"Issue_ID":"451","Description1":"Driver not working",
"Current_Status":"TakenUp","Issue_Logged_Timestamp":"2020-05-07 10:09:42"}]
```

4.4.4 Testcase ID: API-004

Description: Validating Services.dart API call with action '_GET_ALL_ISSUES_BY_EMP', it successfully returns issues logged by an employee from his Employee ID

Result: API call is valid.

4.4.5 Testcase ID: API-005



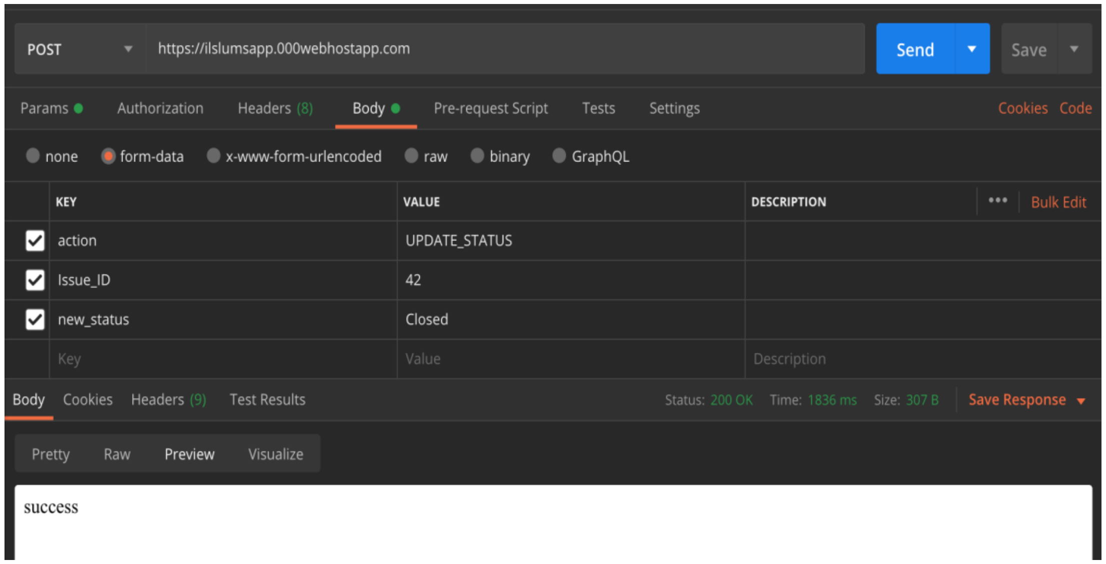
action	_GET_ALL_ISSUES	

Status: 200 OK Time: 1514 ms Size: 4.84 KB

```
[{"Issue_ID":"1","Description1":"Zoom is not installing.","Current_Status":"Closed","Issue_Logged_Timestamp":"2020-04-26 11:26:50"},
{"Issue_ID":"11","Description1":"LCD not working","Current_Status":"Fixed","Issue_Logged_Timestamp":"2020-04-27 20:16:25"},
{"Issue_ID":"21","Description1":"Heating up","Current_Status":"TakenUp","Issue_Logged_Timestamp":"2020-04-27 20:21:32"},
```

Description: Validating Services.dart API call with action '_GET_ALL_ISSUES', it successfully returns all issues from the issues table

Result: API call is valid.



POST https://ilslumsapp.000webhostapp.com

Body: form-data

KEY	VALUE	DESCRIPTION
action	UPDATE_STATUS	
Issue_ID	42	
new_status	Closed	

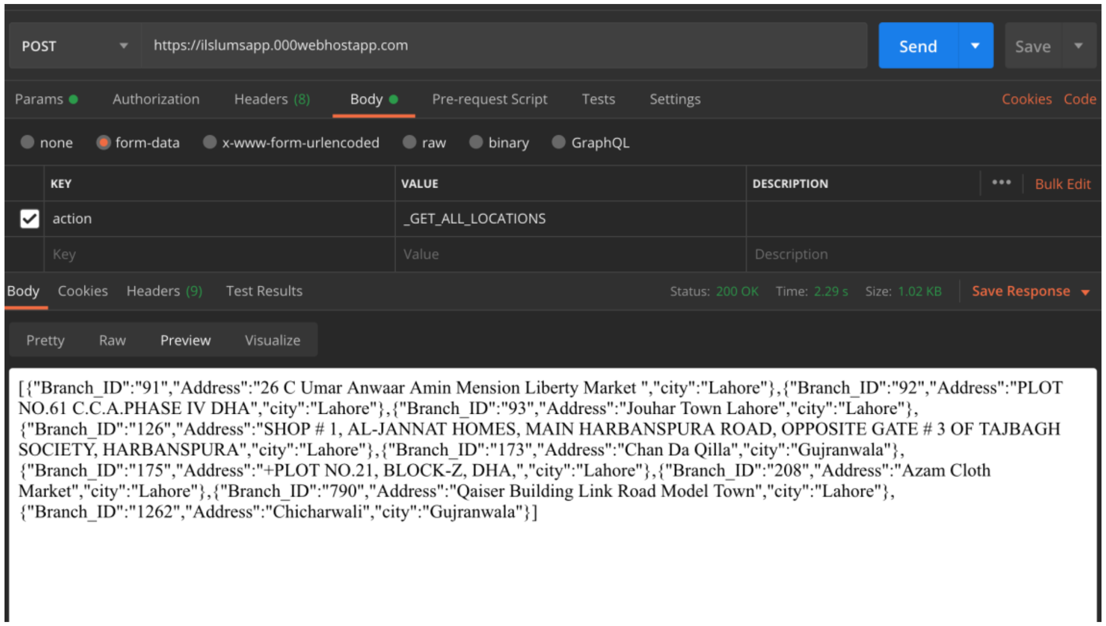
Status: 200 OK Time: 1836 ms Size: 307 B

```
success
```

4.4.6 Testcase ID: API-006

Description: Validating Services.dart API call with action 'UPDATE_STATUS', it updates Current_Status of the issue logged in issues table. API call returns success and Current_Status of issue is updated successfully.

Result: API call is valid.



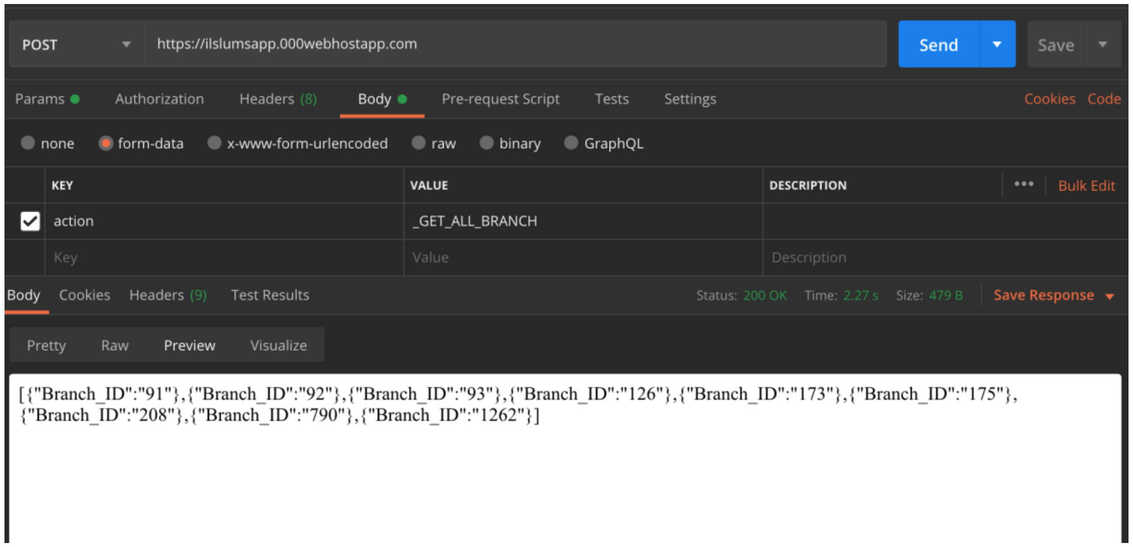
KEY	VALUE	DESCRIPTION
✓ action	_GET_ALL_LOCATIONS	
Key	Value	Description

```
[{"Branch_ID":"91","Address":"26 C Umar Anwaar Amin Mension Liberty Market ","city":"Lahore"},{"Branch_ID":"92","Address":"PLOT
NO.61 C.C.A.PHASE IV DHA","city":"Lahore"},{"Branch_ID":"93","Address":"Jouhar Town Lahore","city":"Lahore"},
{"Branch_ID":"126","Address":"SHOP # 1, AL-JANNAT HOMES, MAIN HARBANSPURA ROAD, OPPOSITE GATE # 3 OF TAJBAGH
SOCIETY, HARBANSPURA","city":"Lahore"},{"Branch_ID":"173","Address":"Chan Da Qilla","city":"Gujranwala"},
{"Branch_ID":"175","Address":"+PLOT NO.21, BLOCK-Z, DHA,","city":"Lahore"},{"Branch_ID":"208","Address":"Azam Cloth
Market","city":"Lahore"},{"Branch_ID":"790","Address":"Qaiser Building Link Road Model Town","city":"Lahore"},
{"Branch_ID":"1262","Address":"Chicharwali","city":"Gujranwala"}]
```

4.4.7 Testcase ID: API-007

Description: Validating Services.dart API call with action '_GET_ALL_LOCATIONS', it successfully returns list of all branches details from the branch table.

Result: API call is valid.



KEY	VALUE	DESCRIPTION
✓ action	_GET_ALL_BRANCH	

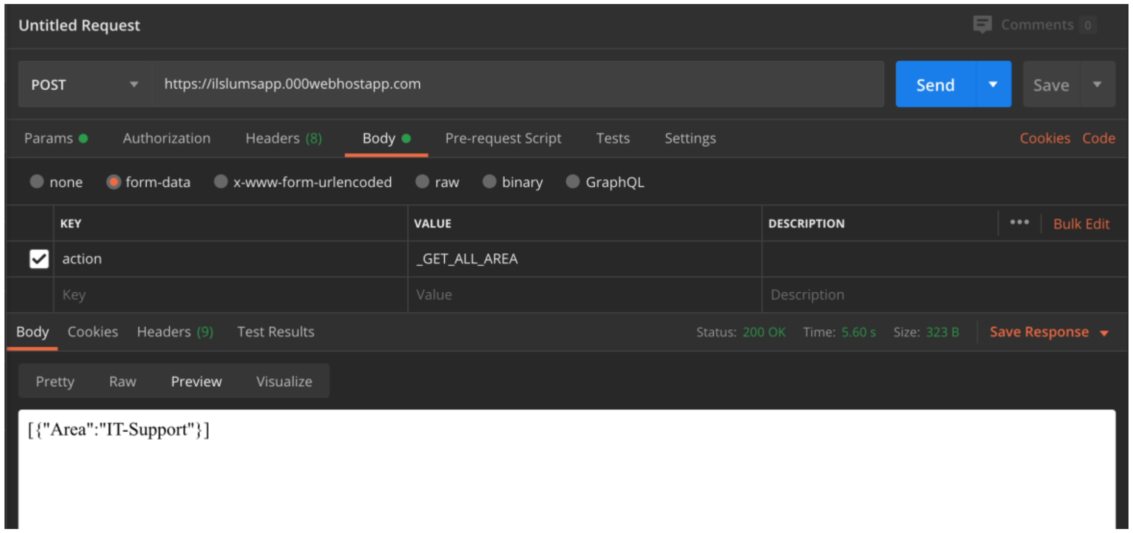
Status: 200 OK Time: 2.27 s Size: 479 B

```
[{"Branch_ID":"91"},{"Branch_ID":"92"},{"Branch_ID":"93"},{"Branch_ID":"126"},{"Branch_ID":"173"},{"Branch_ID":"175"},
{"Branch_ID":"208"},{"Branch_ID":"790"},{"Branch_ID":"1262"}]
```

4.4.8 Testcase ID: API-008

Description: Validating Services.dart API call with action '_GET_ALL_BRANCH', it successfully returns list of all distinct Branch_IDs from the branch table.

Result: API call is valid.



POST https://ilslumsapp.000webhostapp.com

KEY	VALUE	DESCRIPTION
action	_GET_ALL_AREA	

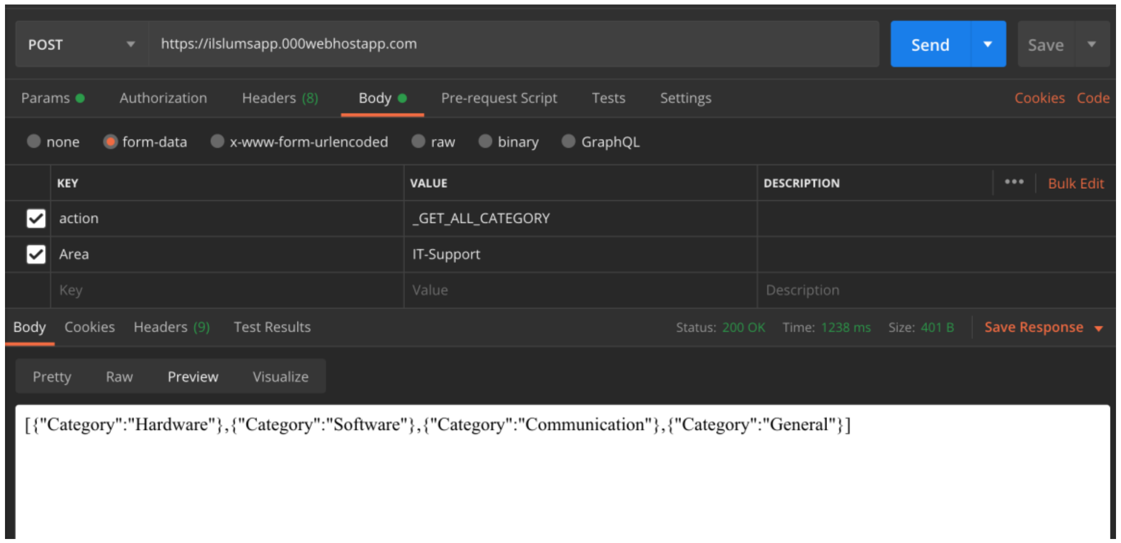
Status: 200 OK Time: 5.60 s Size: 323 B

```
[{"Area":"IT-Support"}]
```

4.4.9 Testcase ID: API-009

Description: Validating Services.dart API call with action '_GET_ALL_AREA', it successfully returns list of all unique Area of Expertise from the support_team table.

Result: API call is valid.



https://ilslumsapp.000webhostapp.com

KEY	VALUE	DESCRIPTION
action	_GET_ALL_CATEGORY	
Area	IT-Support	
Key	Value	Description

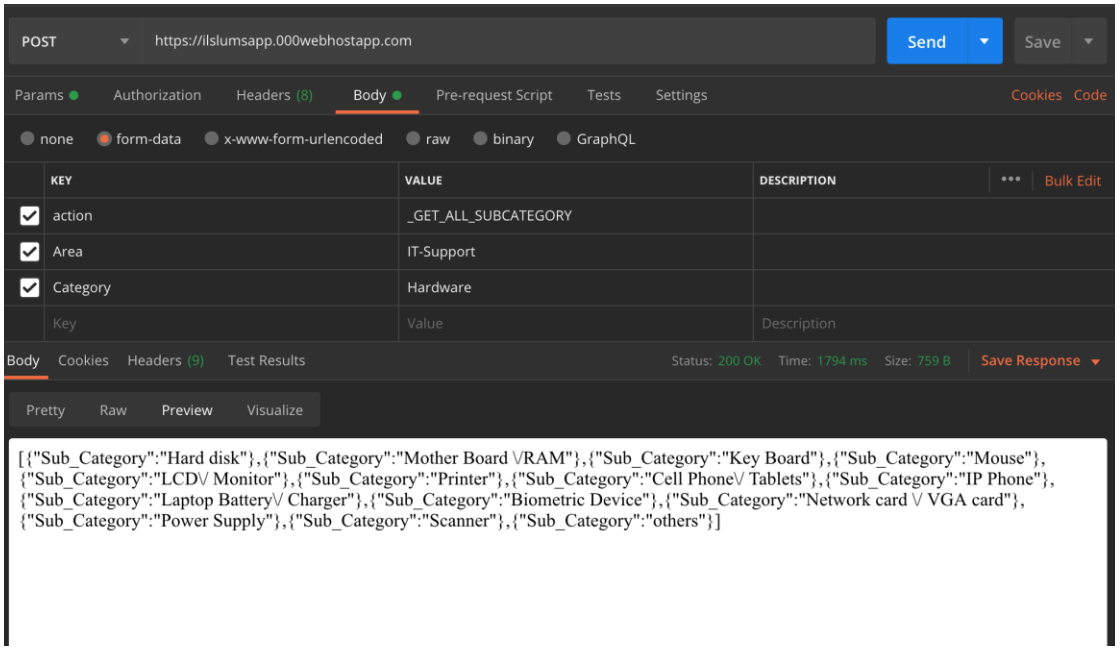
Status: 200 OK Time: 1238 ms Size: 401 B

```
[{"Category":"Hardware"},{"Category":"Software"},{"Category":"Communication"},{"Category":"General"}]
```

4.4.10 Testcase ID: API-010

Description: Validating Services.dart API call with action '_GET_ALL_CATEGORY', it successfully returns list of all unique categories of support_team member expertise from the support_team table

Result: API call is valid.



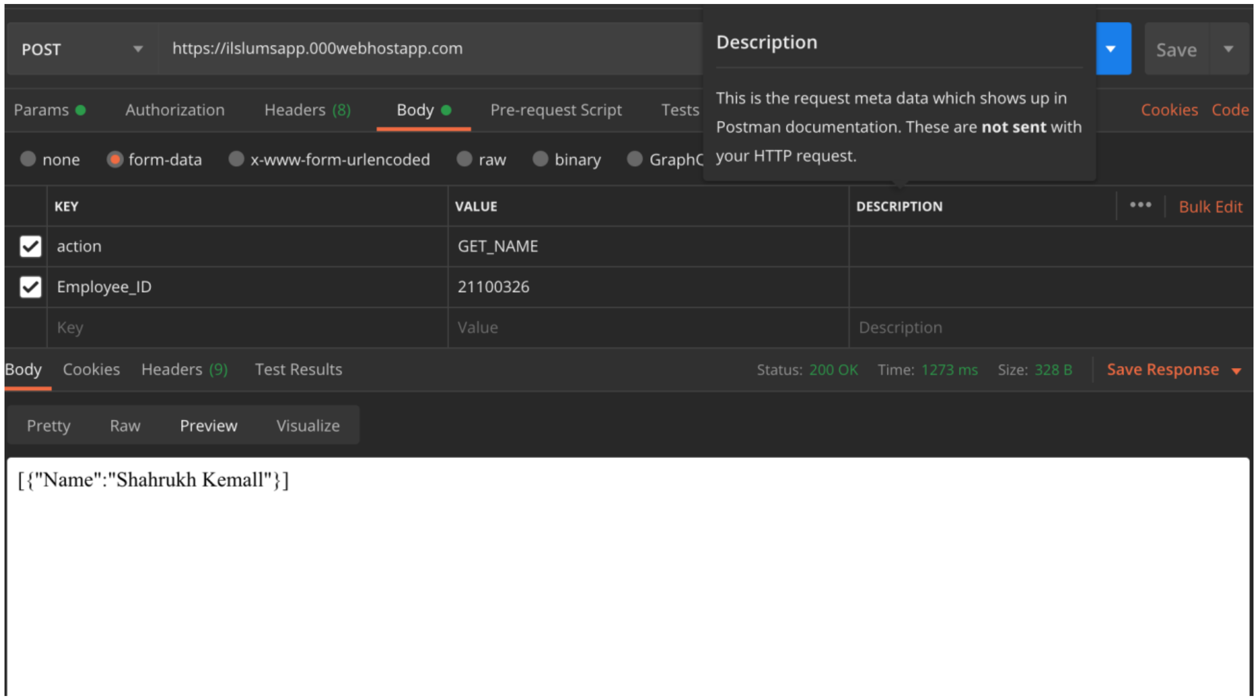
KEY	VALUE	DESCRIPTION
✓ action	_GET_ALL_SUBCATEGORY	
✓ Area	IT-Support	
✓ Category	Hardware	
Key	Value	Description

```
[{"Sub_Category":"Hard disk"},{"Sub_Category":"Mother Board \/RAM"},{"Sub_Category":"Key Board"},{"Sub_Category":"Mouse"},
{"Sub_Category":"LCD\/ Monitor"},{"Sub_Category":"Printer"},{"Sub_Category":"Cell Phone\/ Tablets"},{"Sub_Category":"IP Phone"},
{"Sub_Category":"Laptop Battery\/ Charger"},{"Sub_Category":"Biometric Device"},{"Sub_Category":"Network card \/ VGA card"},
{"Sub_Category":"Power Supply"},{"Sub_Category":"Scanner"},{"Sub_Category":"others"}]
```

4.4.11 Testcase ID: API-011

Description: Validating Services.dart API call with action '_GET_ALL_SUBCATEGORY', it returns list of all unique sub-categories of support_team member's expertise and category from the support_team table

Result: API call is valid.



This is the request meta data which shows up in Postman documentation. These are **not sent** with your HTTP request.

	KEY	VALUE	DESCRIPTION
✓	action	GET_NAME	
✓	Employee_ID	21100326	
	Key	Value	Description

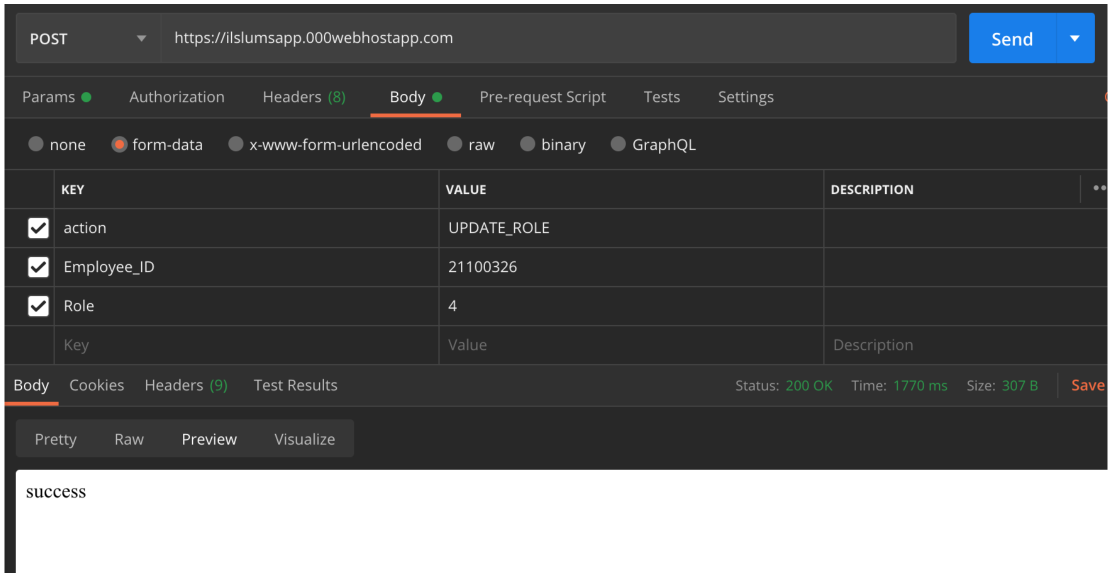
Status: 200 OK Time: 1273 ms Size: 328 B

```
[{"Name":"Shahrukh Kemall"}]
```

4.4.12 Testcase ID: API-012

Description: Validating Services.dart API call with action ‘GET_NAME’, it successfully returns name of the employee against an Employee ID from the employee table.

Result: API call is valid.



https://ilslumsapp.000webhostapp.com

KEY	VALUE	DESCRIPTION
action	UPDATE_ROLE	
Employee_ID	21100326	
Role	4	
Key	Value	Description

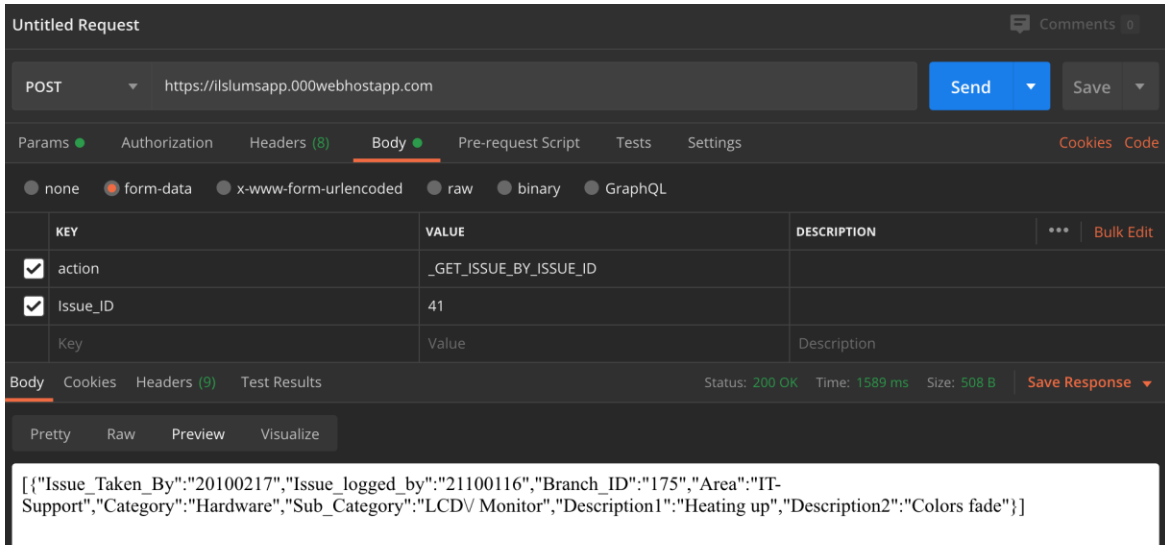
Status: 200 OK Time: 1770 ms Size: 307 B

```
success
```

4.4.13 Testcase ID: API-013

Description: Validating Services.dart API call with action 'GET_ROLE', it successfully returns role of the employee against an Employee ID

Result: API call is valid.



POST https://ilslumsapp.000webhostapp.com

KEY	VALUE	DESCRIPTION
✓ action	_GET_ISSUE_BY_ISSUE_ID	
✓ Issue_ID	41	
Key	Value	Description

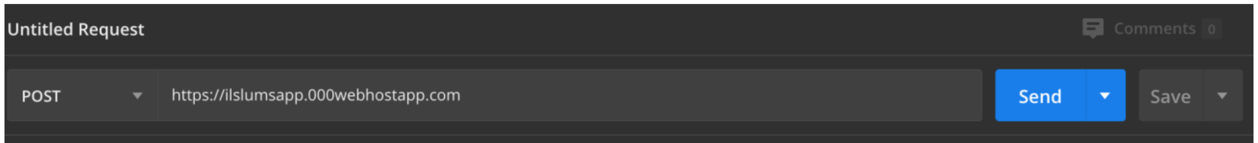
Status: 200 OK Time: 1589 ms Size: 508 B

```
[{"Issue_Taken_By":"20100217","Issue_logged_by":"21100116","Branch_ID":"175","Area":"IT-
Support","Category":"Hardware","Sub_Category":"LCD\/ Monitor","Description1":"Heating up","Description2":"Colors fade"}]
```

4.4.14 Testcase ID: API-014

Description: Validating Services.dart API call with action '_GET_ISSUE_BY_ISSUE_ID', it successfully returns details of the issue against an Issue_ID.

Result: API call is valid.

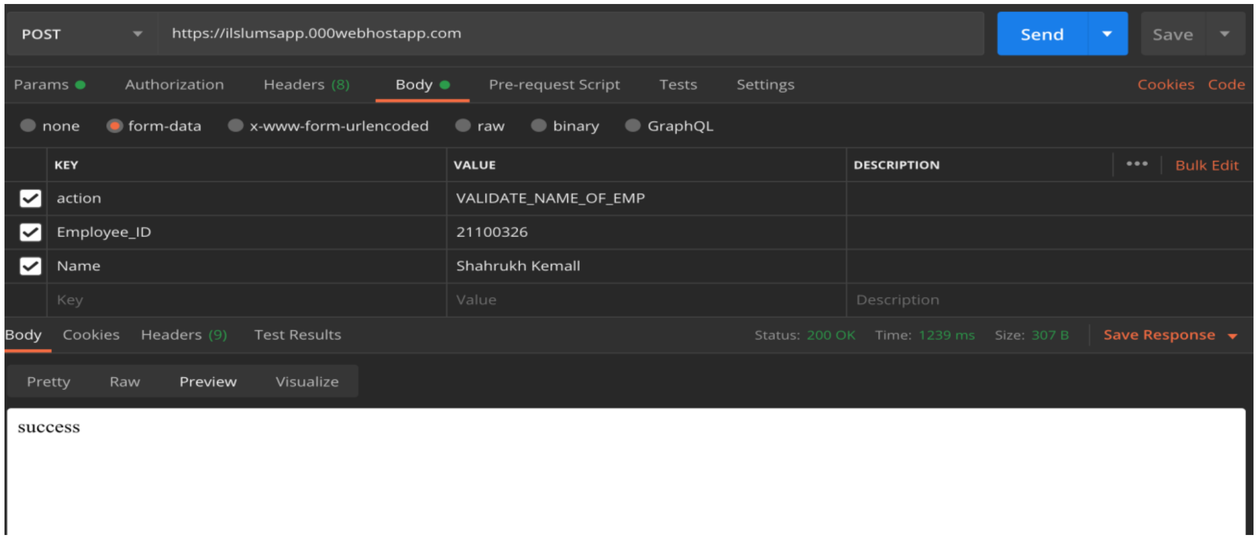


POST https://ilslumsapp.000webhostapp.com

4.4.15 Testcase ID: API-015

Description: Validating Services.dart API call with action '_GET_ALL_LOCATIONS_BY_ID', it returns branch details against a Branch_IDs from the branch table

Result: API call is valid.



https://ilslumsapp.000webhostapp.com

KEY	VALUE	DESCRIPTION
action	VALIDATE_NAME_OF_EMP	
Employee_ID	21100326	
Name	Shahrukh Kemall	

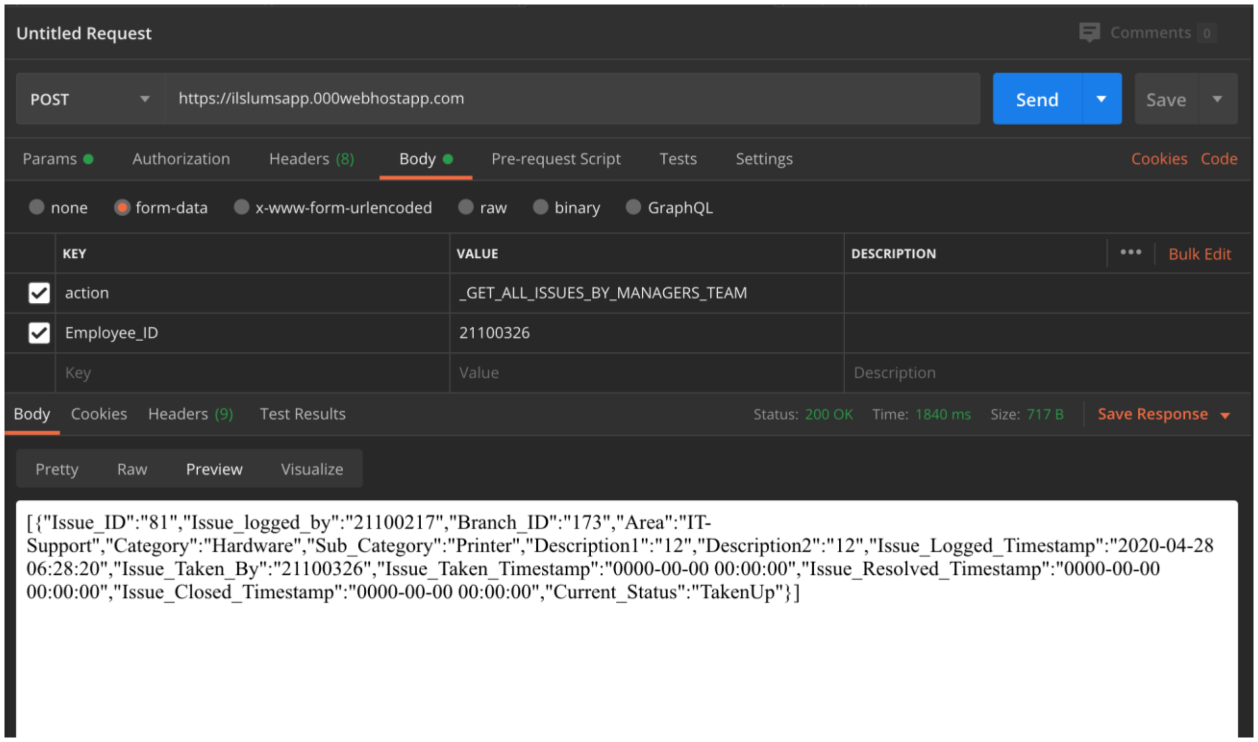
Status: 200 OK Time: 1239 ms Size: 307 B

```
success
```

4.4.16 Testcase ID: API-016

Description: Validating Services.dart API call with action 'VALIDATE_NAME_OF_EMP', it validates an employee ID against the name of the employee in the employee table. It successfully returns success if Employee ID against a Name is correct, and returns Failed otherwise

Result: API call is valid.



POST https://ilslumsapp.000webhostapp.com

Body: form-data

KEY	VALUE	DESCRIPTION
✓ action	_GET_ALL_ISSUES_BY_MANAGERS_TEAM	
✓ Employee_ID	21100326	
Key	Value	Description

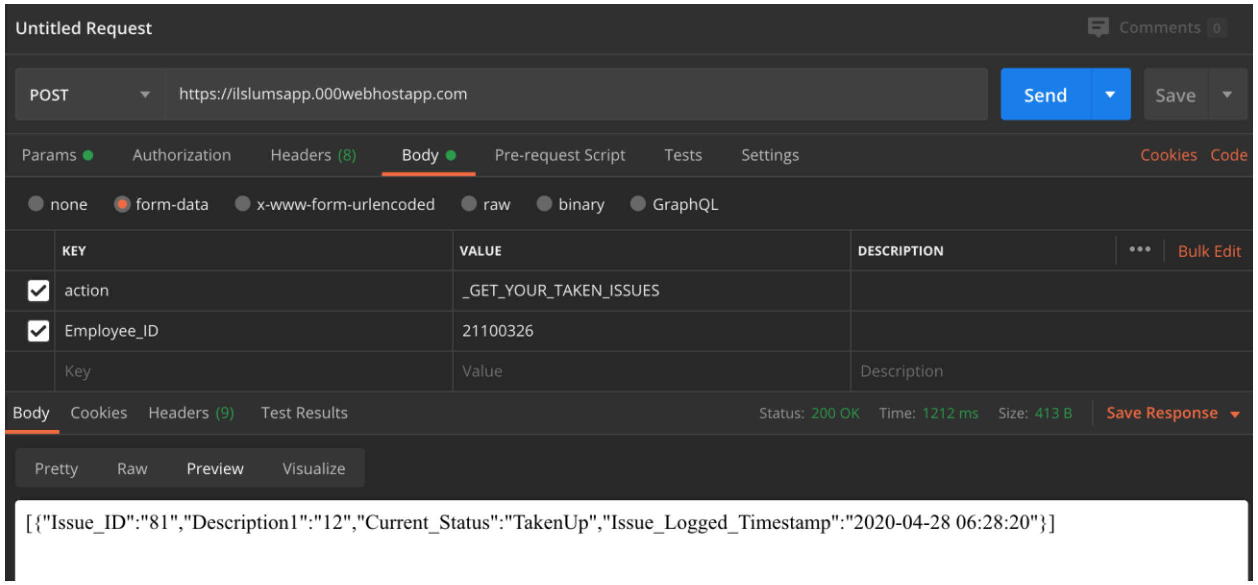
Status: 200 OK Time: 1840 ms Size: 717 B

```
[{"Issue_ID":"81","Issue_logged_by":"21100217","Branch_ID":"173","Area":"IT-Support","Category":"Hardware","Sub_Category":"Printer","Description1":"12","Description2":"12","Issue_Logged_Timestamp":"2020-04-28 06:28:20","Issue_Taken_By":"21100326","Issue_Taken_Timestamp":"0000-00-00 00:00:00","Issue_Resolved_Timestamp":"0000-00-00 00:00:00","Issue_Closed_Timestamp":"0000-00-00 00:00:00","Current_Status":"TakenUp"}]
```

4.4.17 Testcase ID: API-017

Description: Validating Services.dart API call with action '_GET_ALL_ISSUES_BY_MANAGERS_TEAM', it successfully returns issues taken up by a manager's team.

Result: API call is valid.



POST https://ilslumsapp.000webhostapp.com

	KEY	VALUE	DESCRIPTION
✓	action	_GET_YOUR_TAKEN_ISSUES	
✓	Employee_ID	21100326	
	Key	Value	Description

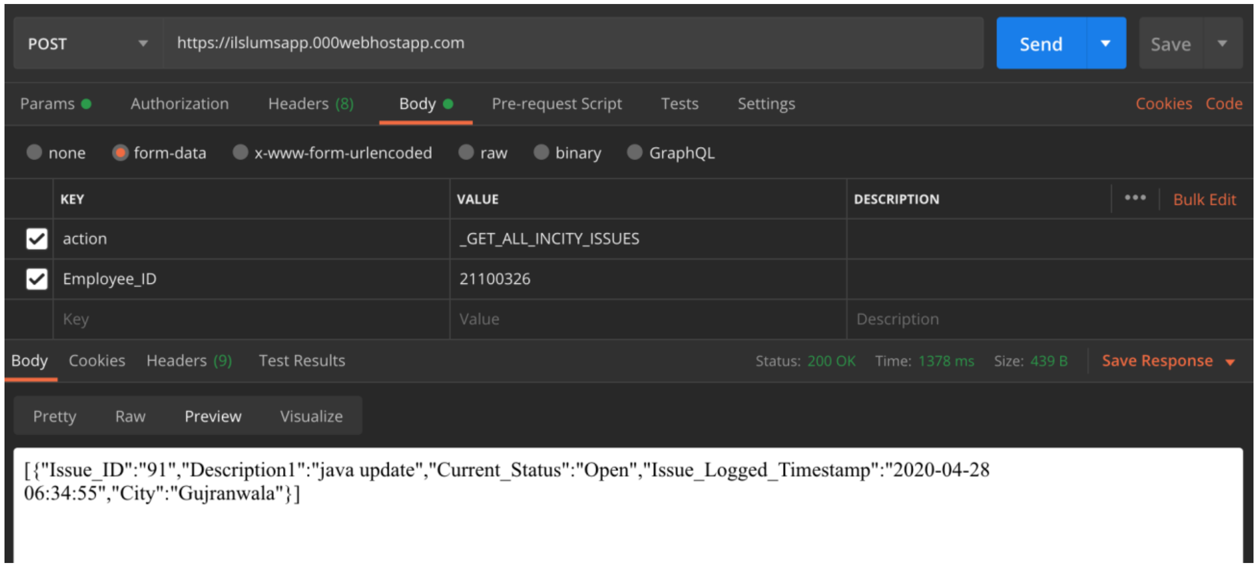
Status: 200 OK Time: 1212 ms Size: 413 B

```
[{"Issue_ID":"81","Description1":"12","Current_Status":"TakenUp","Issue_Logged_Timestamp":"2020-04-28 06:28:20"}]
```

4.4.18 Testcase ID: API-018

Description: Validating Services.dart API call with action '_GET_YOUR_TAKEN_ISSUES', it successfully returns details of issues taken up by a Support Team member

Result: API call is valid.



https://ilslumsapp.000webhostapp.com

KEY	VALUE	DESCRIPTION
action	_GET_ALL_INCITY_ISSUES	
Employee_ID	21100326	

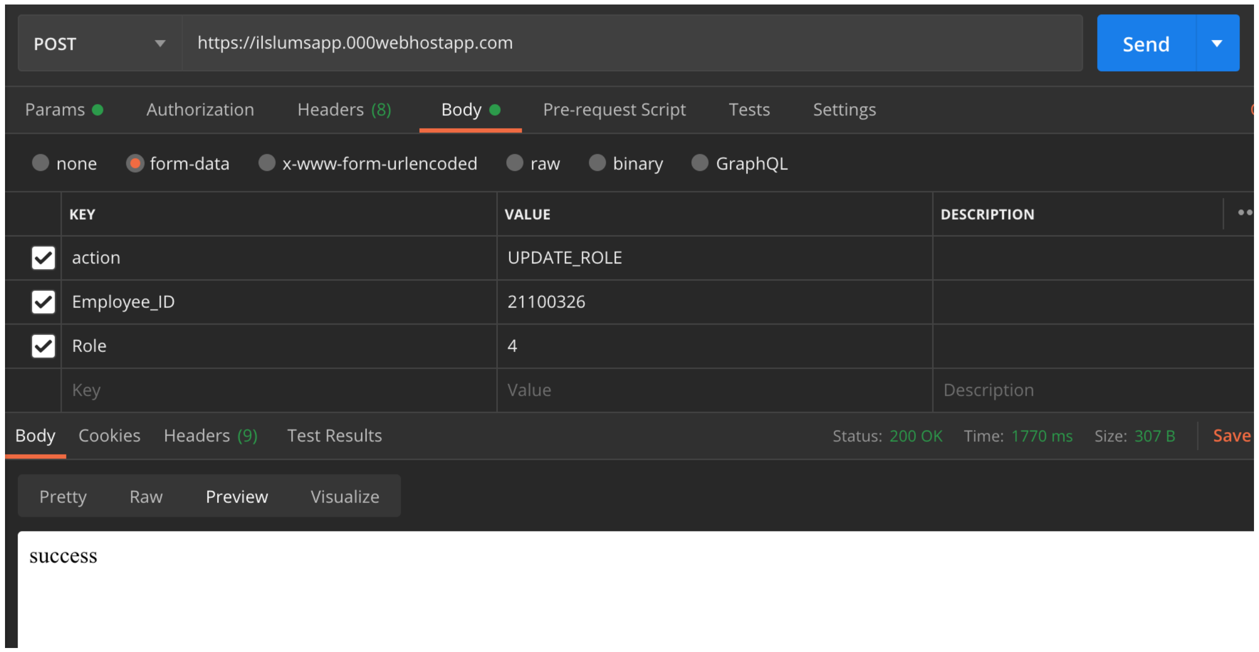
Status: 200 OK Time: 1378 ms Size: 439 B

```
[{"Issue_ID":"91","Description1":"java update","Current_Status":"Open","Issue_Logged_Timestamp":"2020-04-28
06:34:55","City":"Gujranwala"}]
```

4.4.19 Testcase ID: API-019

Description: Validating Services.dart API call with action ‘_GET_ALL_INCITY_ISSUES’, it successfully returns all relevant issues with respect to support_team member’s location (city).

Result: API call is valid.



https://ilslumsapp.000webhostapp.com

KEY	VALUE	DESCRIPTION
action	UPDATE_ROLE	
Employee_ID	21100326	
Role	4	
Key	Value	Description

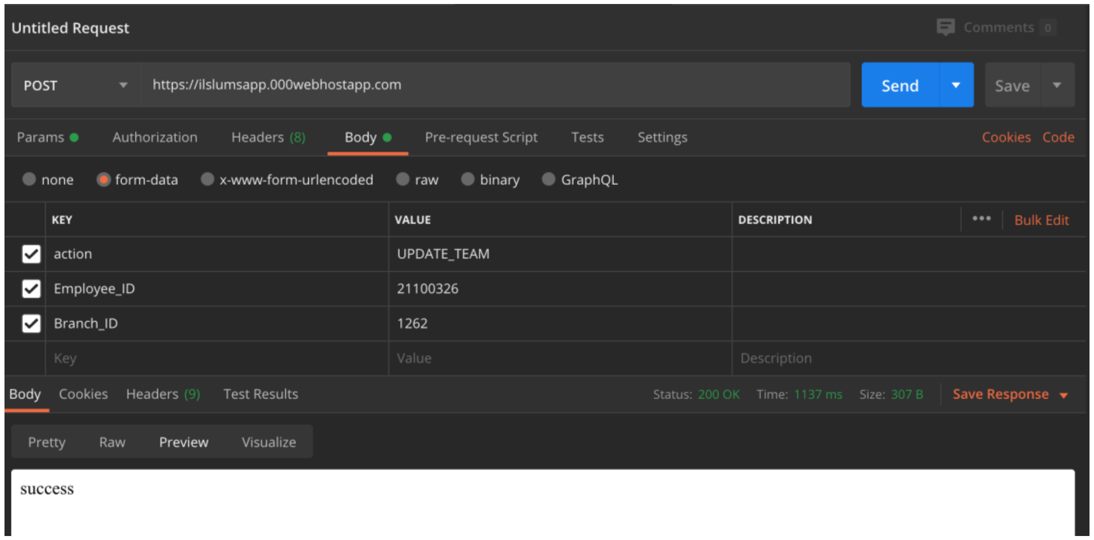
Status: 200 OK Time: 1770 ms Size: 307 B

```
success
```

4.4.20 Testcase ID: API-020

Description: Validating Services.dart API call with action ‘UPDATE_ROLE’, it successfully updates role of the employee against its Employee ID.

Result: API call is valid.



POST https://ilslumsapp.000webhostapp.com

none [✓ form-data] x-www-form-urlencoded raw binary GraphQL

KEY	VALUE	DESCRIPTION
✓ action	UPDATE_TEAM	
✓ Employee_ID	21100326	
✓ Branch_ID	1262	
Key	Value	Description

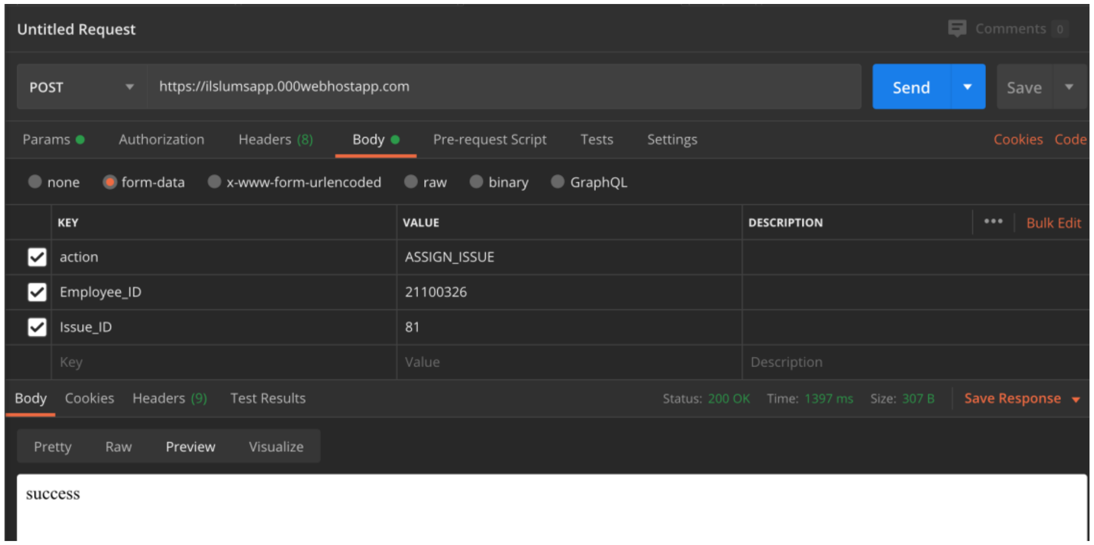
Status: 200 OK Time: 1137 ms Size: 307 B

```
success
```

4.4.21 Testcase ID: API-021

Description: Validating Services.dart API call with action ‘UPDATE_TEAM’, it changes team/Branch of the employee against its Employee ID

Result: API call is valid.


Untitled Request

POST https://ilslumsapp.000webhostapp.com

Body: form-data

KEY	VALUE	DESCRIPTION
action	ASSIGN_ISSUE	
Employee_ID	21100326	
Issue_ID	81	

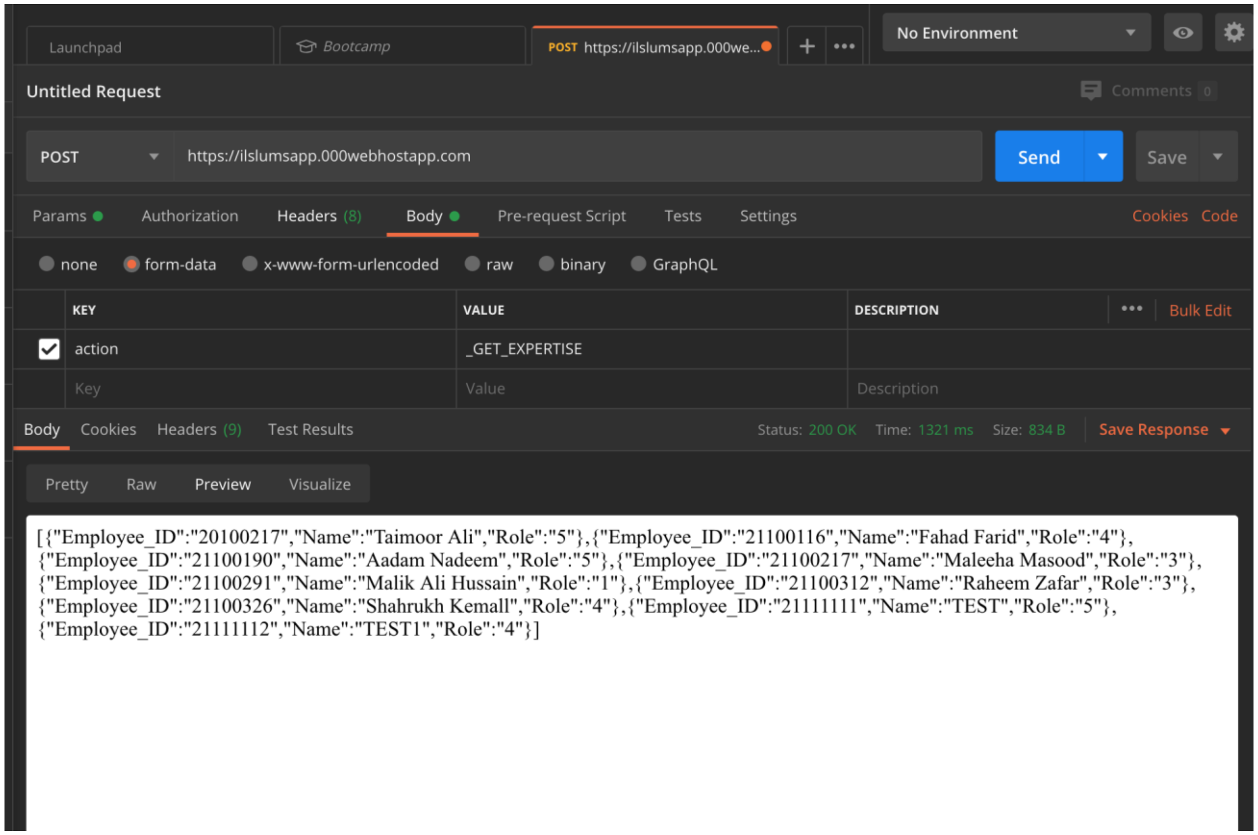
Status: 200 OK Time: 1397 ms Size: 307 B

```
success
```

4.4.22 Testcase ID: API-022

Description: Validating Services.dart API call with action ‘ASSIGN_ISSUE’, it assigns an issue to an Employee ID by successfully updating issue_taken_up field in issues table

Result: API call is valid.



POST https://ilslumsapp.000webhostapp.com

KEY	VALUE	DESCRIPTION
action	_GET_EXPERTISE	

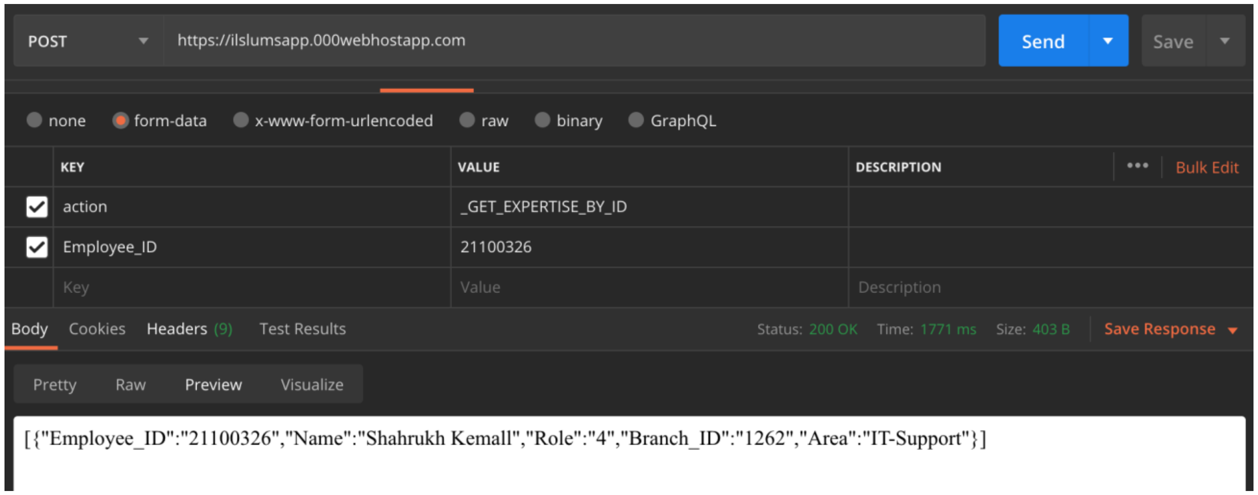
Status: 200 OK Time: 1321 ms Size: 834 B

```
[{"Employee_ID":"20100217","Name":"Taimoor Ali","Role":"5"},{"Employee_ID":"21100116","Name":"Fahad Farid","Role":"4"},
{"Employee_ID":"21100190","Name":"Aadam Nadeem","Role":"5"},{"Employee_ID":"21100217","Name":"Maleeha Masood","Role":"3"},
{"Employee_ID":"21100291","Name":"Malik Ali Hussain","Role":"1"},{"Employee_ID":"21100312","Name":"Raheem Zafar","Role":"3"},
{"Employee_ID":"21100326","Name":"Shahrukh Kemall","Role":"4"},{"Employee_ID":"21111111","Name":"TEST","Role":"5"},
{"Employee_ID":"21111112","Name":"TEST1","Role":"4"}]
```

4.4.23 Testcase ID: API-023

Description: Validating Services.dart API call with action '_GET_EXPERTISE', it successfully returns details (Name, Employee_ID, Role) of all employees

Result: API call is valid.



POST https://ilslumsapp.000webhostapp.com

Send Save

none [✓ form-data] x-www-form-urlencoded raw binary GraphQL

	KEY	VALUE	DESCRIPTION
✓	action	_GET_EXPERTISE_BY_ID	
✓	Employee_ID	21100326	
	Key	Value	Description

Body Cookies Headers (9) Test Results Status: 200 OK Time: 1771 ms Size: 403 B Save Response

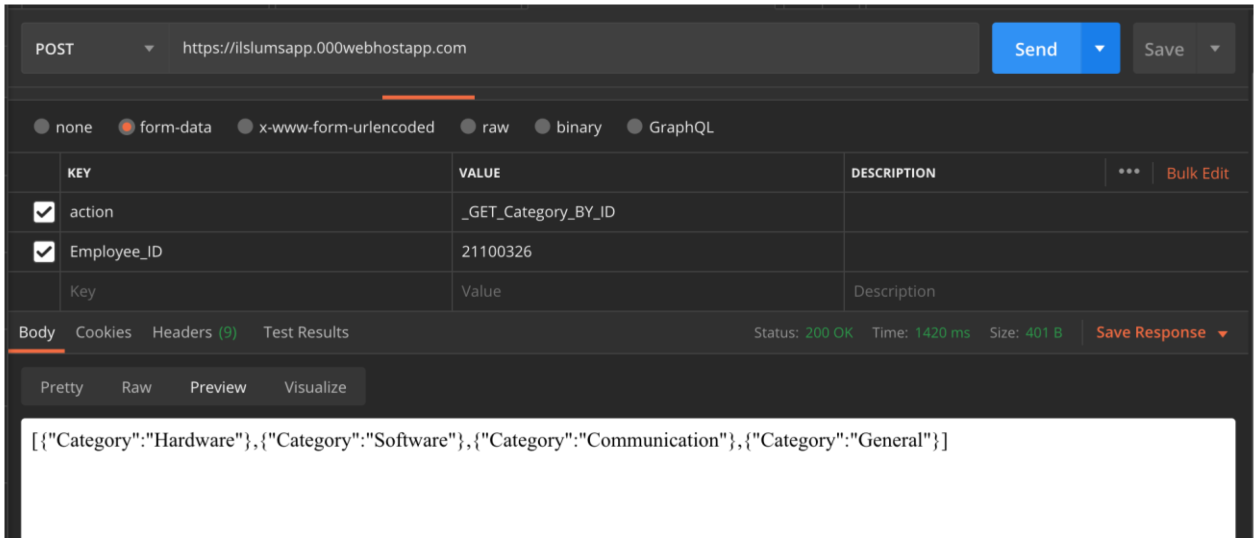
Pretty Raw Preview Visualize

```
[{"Employee_ID":"21100326","Name":"Shahrukh Kemall","Role":"4","Branch_ID":"1262","Area":"IT-Support"}]
```

4.4.24 Testcase ID: API-024

Description: Validating Services.dart API call with action ‘_GET_EXPERTISE_BY_ID’, it successfully returns details (Name, Employee_ID, Role) of an Employee by its Employee ID

Result: API call is valid.


POST https://ilslumsapp.000webhostapp.com

Body: form-data

KEY	VALUE	DESCRIPTION
action	_GET_Category_BY_ID	
Employee_ID	21100326	

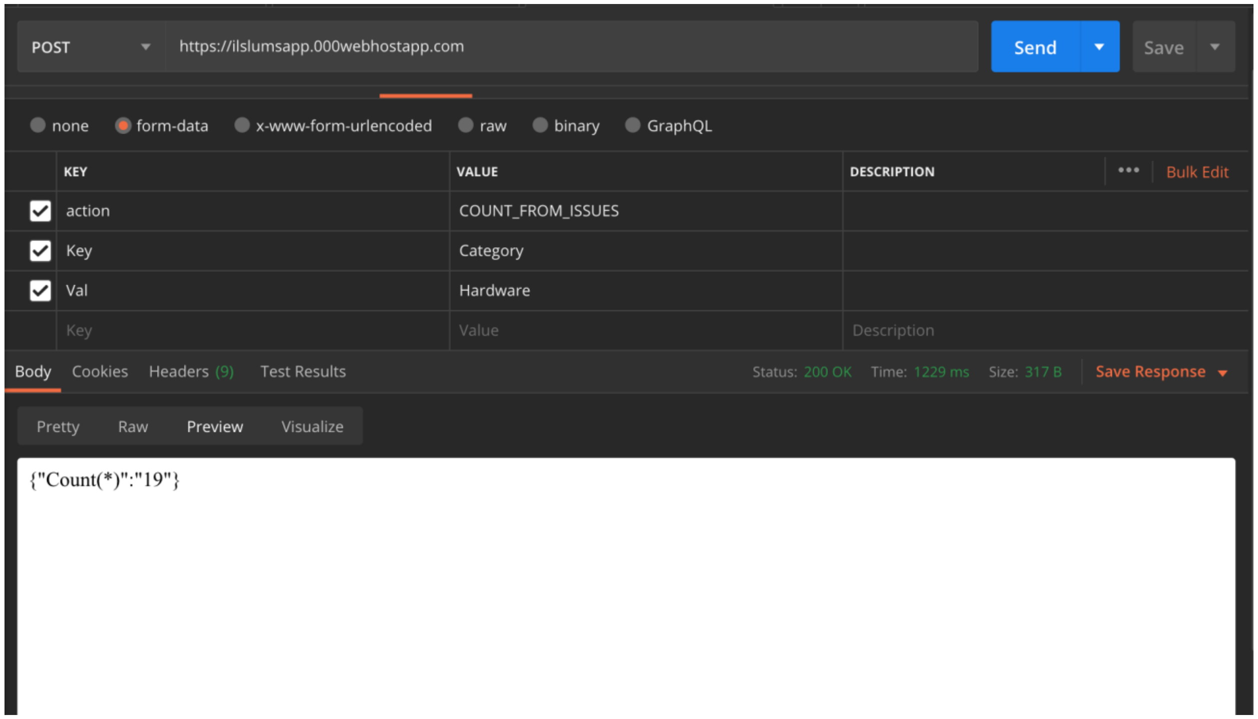
Status: 200 OK Time: 1420 ms Size: 401 B

```
[{"Category":"Hardware"},{"Category":"Software"},{"Category":"Communication"},{"Category":"General"}]
```

4.4.25 Testcase ID: API-025

Description: Validating Services.dart API call with action '_GET_Category_BY_ID', it successfully returns all distinct categories in an area of expertise of a Support Team Member

Result: API call is valid.



POST https://ilslumsapp.000webhostapp.com

form-data

KEY	VALUE	DESCRIPTION
action	COUNT_FROM_ISSUES	
Key	Category	
Val	Hardware	

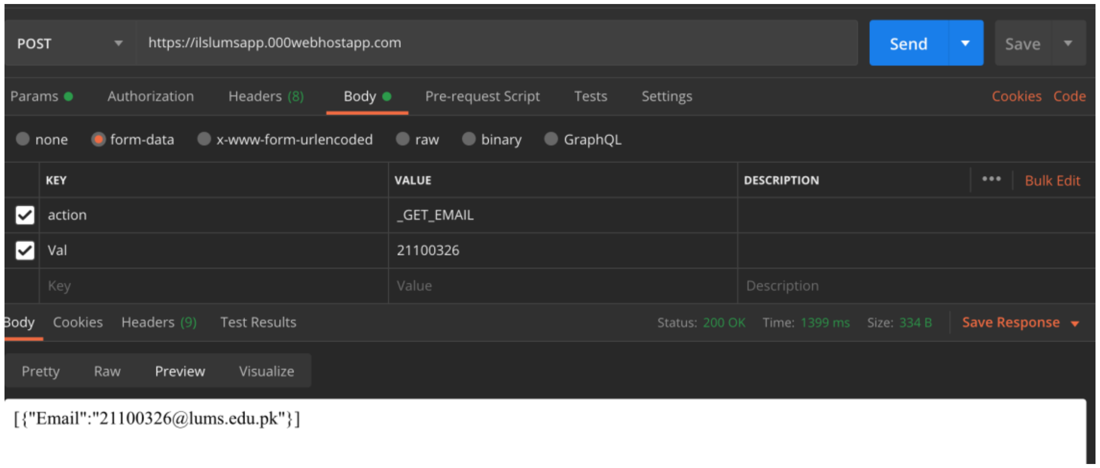
Status: 200 OK Time: 1229 ms Size: 317 B

```
{"Count(*)":"19"}
```

4.4.26 Testcase ID: API-026

Description: Validating Services.dart API call with action ‘COUNT_FROM_ISSUES’, it returns number of issues logged in issues table where the ‘key’ column contains the ‘val’ value, both provided according to the need.

Result: API call is valid.



KEY	VALUE	DESCRIPTION
✓ action	_GET_EMAIL	
✓ Val	21100326	
Key	Value	Description

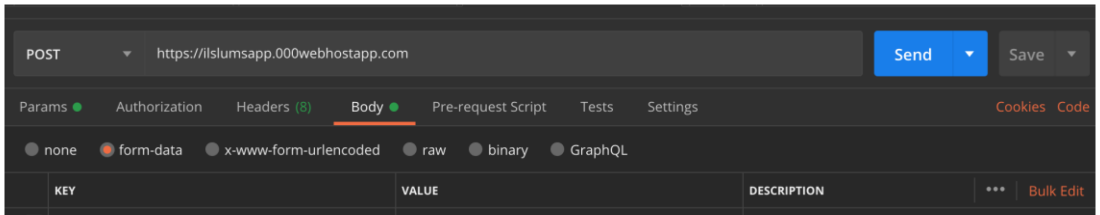
```
[{"Email":"21100326@lums.edu.pk"}]
```

4.4.27 Testcase ID: API-027

Description: Validating Services.dart API call with action '_GET_EMAIL', it successfully returns email stored against an Employee ID in employee table

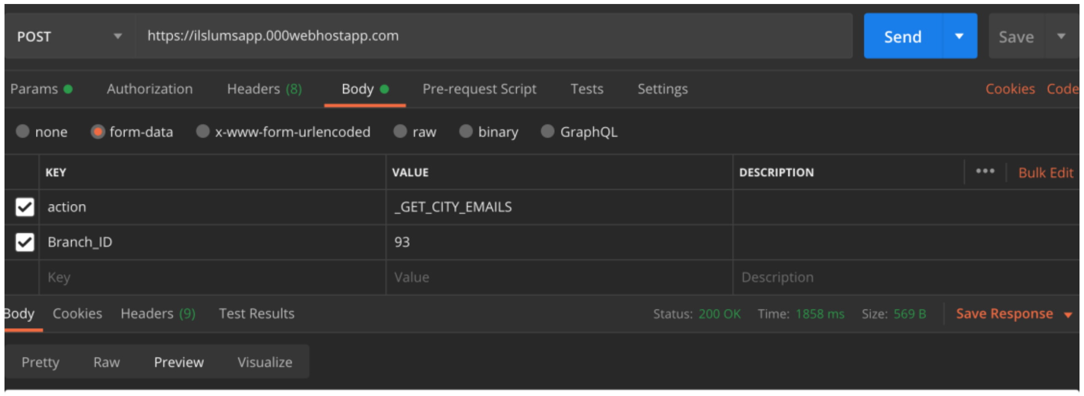
Result: API call is valid.

4.4.28 Testcase ID: API-028



Description: Validating Services.dart API call with action '_GET_Team_EMAILS', it successfully returns list of emails of relevant support team members when issue within their team/Branch ID and category of their area of expertise is logged.

Result: API call is valid.



https://ilslumsapp.000webhostapp.com

KEY	VALUE	DESCRIPTION
action	_GET_CITY_EMAILS	
Branch_ID	93	
Key	Value	Description

Status: 200 OK Time: 1858 ms Size: 569 B

```
[{"Email":"21100312@lums.edu.pk"},{"Email":"21100291@lums.edu.pk"},{"Email":"21100217@lums.edu.pk"},
{"Email":"21100190@lums.edu.pk"},{"Email":"fahadfarid@gmail.com"},{"Email":"taimoorali@gmail.com"},
{"Email":"21100326@lums.edu.pk"},{"Email":"shahrukhkemall@gmail.com"}]
```

4.4.29 Testcase ID: API-029

Description: Validating Services.dart API call with action '_GET_CITY_EMAILS', it successfully returns list of emails of relevant support team members when issue within their city and category of their area of expertise is logged

Result: API call is valid.

5. Automated testing – Maleeha Masood

The purpose of performing automated testing was to clearly monitor the responses from the database without going through the hassle of inserting extra print statements within the code to view fetched results.

The scope of the testing is limited to functions that retrieve data from the database. This is because the kind of unit testing, in the context of Flutter apps meaning testing an individual function, done by using the Flutter Test Package the way we did made the testing of the functionality of the database very easy. Other functions used throughout the app were not suitable to be tested this way. Using the Testing Package for widget testing was not suitable given the way our app was structured since it requires running the app in real time for variables that the UI uses to place widgets on the screen. This was not possible with the Flutter Testing Package.

5.1 List of test cases used for automated testing

Testcase ID	Testcase name	Performed by	Successful/Failed
Module: Database			
A001	valEmployee Function	Maleeha Masood	Successful
A002	getRole Function	Maleeha Masood	Successful
A003	valEmployeeId Function	Maleeha Masood	Successful

5.2 Flutter Testing Package

Flutter Testing Package: The main reason of picking this tool for testing is because of its ease of use with flutter apps- no additional tool is necessary and a basic coding setup like VS Code is good to go. This was especially helpful because the simple, but core functionality of the database has been tested by using this tool which has helped reduced the complexity that the manual testing of the database would have caused.

5.3 Setup

Only a simple coding setup is needed- VS Code with its flutter and dart plugins was used for the purpose of this document. Dart knowledge is also necessary. No extra installation is needed.

5.4 Test Case A001: valEmployee Function - Maleeha Masood

5.4.1 Description

When a user tries to login to their account, they enter their Employee ID and Password. The employee ID is used to query the database and the corresponding password in the database is retrieved. The entered password is compared to the retrieved one to validate the user. valEmployee function is used to validate the entered inputs. This function is tested in this test case.

5.4.2 Pre-Requisites

Text Editor (VS Code used for this test case), code repository locally stored, Dart programming language knowledge, terminal and valid employee ID and password pairs.

5.4.3 Script Used

```dart
import 'package:flutter_test/flutter_test.dart';
import 'package:ils/services.dart';

void main() {
  test('Correct Fetching from Database', (){
    var employees = ["21100217", "21100291", "21100190", "21100312", "21100326"];
    var passwords = ["abc21100217", "abc21100291", "abc21100190", "abc21100312", "abc21100326"];
    for (int i = 0; i < 5; i++) {
      String employee = employees[i];
      String password = passwords[i];
      AppServices.valEmployee(employee, password).then((result){expect(result, 'success'); print('success');});
    }
    var incorpasswords = ["21100217", "21100291", "21100190", "21100312", "21100326"];
    for (int i = 0; i < 5; i++) {
      String employee = employees[i];
      String password = incorpasswords[i];
      AppServices.valEmployee(employee, password).then((result){expect(result, 'error'); print('success');});
    }
  });
}
```

5.4.4 Description of the Script

The script tests the functioning of the valEmployee function that takes in an employee ID and password pair and returns 'success' if they match the one stored in the database. Otherwise, 'error' is returned.
In this script, 5 valid usernames and their passwords are stored in an array. Then, each pair of employee ID and password are sent to the valEmployee function in which the employee ID is queried from the database. The results of the query are compared to the passwords in the pair. If these match, then success is returned by the function. If all 5 pairs result in success, then this part of the test passes.
The next part of the script creates a new array of incorrect passwords to create 5 pair of valid employee IDs but incorrect passwords. This time, the results of the queried employee ID are compared to an incorrect password and if they don't match, then the test passes.

5.4.5 Result

Test is Successful.

5.5 Test Case A002: getRole Function - Maleeha Masood

5.5.1 Description

When a user tries to login to their account, their employee ID is used to query their role from the database. The getRole function facilitates this. This function is tested in this test case.

5.5.2 Pre-Requisites

Text Editor (VS Code used for this test case), code repository locally stored, Dart programming language knowledge, terminal and Test Case 0002.

5.5.3 Script Used

```
import 'package:flutter_test/flutter_test.dart';
import 'package:ils/services.dart';

void main() {
  test('Correct Fetching from Database', (){
    var employees = ["21100217", "21100291", "21100190", "21100312", "21100326"];
    var roles = ["Manager", "Admin", "Initiator", "Monitor", "Support"];
    for (int i = 0; i < 5; i++) {
      String employee = employees[i];
      AppServices.getRole(employee).then((result){expect(result, roles[i]); print('success');});
    }
  });
}
```

5.5.4 Description of the Script

The script tests the functioning of the getRole function that takes in an employee ID and returns its role. We compare the returned role and the role that we know is correct for 5 different valid employee IDs. If both these values match, then the test passes.

5.5.5 Result

Test is Successful.

5.6 Test Case A003: valEmployeeId Function - Maleeha Masood

5.6.1 Description

When a user tries to log an issue on someone's behalf, then the user has to enter a valid Employee ID to proceed. valEmployeeId checks the validity of the entered employee ID by using it to query the database.

5.6.2 Pre-Requisites

Text Editor (VS Code used for this test case), code repository locally stored, Dart programming language knowledge, terminal, valid employee IDs and Test Case LOGANISSUE.

5.6.3 Script Used

```
import 'package:flutter_test/flutter_test.dart';
import 'package:ils/services.dart';

void main() {
  test('Correct Fetching from Database', (){
    var employees = ["21100217", "21100291", "21100190", "21100312", "21100326"];
    for (int i = 0; i < 5; i++) {
      String employee = employees[i];
      AppServices.valEmployeeId(employee).then((result){expect(result, employees[i]); print('success');});
    }
    var incoremployees = ["211002171", "211002911", "211001901", "211003121", "211003261"];
    for (int i = 0; i < 5; i++) {
      String employee = incoremployees[i];
      AppServices.valEmployeeId(employee).then((result){expect(result, 'error1'); print('success');});
    }
  });
}
```

5.6.4 *Description of the Script*

The script tests the functioning of the valEmployeeId function that takes in an employee ID and checks for its presence in the database. We compare the results of the function with 5 different valid Employee IDs. If the function finds them in the database, then this part of the test passes.
Next, we compare the results of the function with 5 different invalid Employee IDs. If the function does not find them in the database, then this part of the test also passes.

5.6.5 *Result*

Test is Successful.