Software Requirements Specification Document
(CS360)

Issue Logging System for Allied Bank Limited

Group Number: 24

Aadam Nadeem
Malik Ali Hussain
Maleeha Masood
Muhammad Raheem Zafar
Shahrukh Kemall

Course: Software Engineering CS360

Instructor: Suleman Shahid

University: Lahore University of Management Sciences (LUMS)

Version: 1.0
Date: (24/02/2020)
Number of hours spent on this document: 30 hours

Table of Contents

1 INTRODUCTION 1

1.1 Document Purpose 1
1.2 Product Scope 1
1.3 Intended Audience and Document Overview 1
1.4 Definitions, Acronyms and Abbreviations 2
1.5 References and Acknowledgments 2

2 OVERALL DESCRIPTION 3

2.1 Product Perspective 3
2.2 Product Functionality 4
2.3 Users and Characteristics 6
2.4 Assumptions and Dependencies 8

3 SPECIFIC REQUIREMENTS 8

3.1 Functional Requirements 8
3.2 External Interface Requirements 11
3.2.1 User Interfaces 11
3.2.2 Hardware Interfaces 12
3.2.3 Software Interfaces 12
3.3 Use Case View 13
3.3.1 Use Case Table 13
3.3.2 Use Case Diagram 14
3.3.3 Use Case Description 16

4 OTHER NON-FUNCTIONAL REQUIREMENTS 17

4.1 Performance Requirements 17
4.2 Safety and Security Requirements 18
4.3 Software Quality Attributes 19

Appendix A Top 10 User Stories 21
Appendix B Architectural Spike 21
Appendix C Group Log 21
Appendix D Contribution Statement 23

1 Introduction

1.1 Document Purpose

The purpose of this document is to provide the complete functional, non-functional and interface- related software requirements for the first version of an Issue Logging System mobile application, further referred to in this document as an ILS.
This mobile application is a translation of the web version of the ILS, that is currently used by the employees at Allied Bank Limited, and will be used by employees of Allied Bank Limited to report technical issues with company provided electronic devices.

1.2 Product Scope

The Issue Logging System Application acts as a platform for all the employees of Allied Bank Limited (ABL). Using the ILS app, employees at ABL can report technical concerns, and the relevant support team, technologically educated employees at ABL, can choose to resolve these concerns.
This application will help in more efficient resolution of technical problems since it aims to be an application that is easy to use on the go: the ILS app aims to be a cross platform application that can easily be used on both iOS and Android Mobile Devices by all the relevant personnel involved. This application also facilitates the swift communication between the people involved, aiming to remove any communication related hindrances from the process of resolving a technical issue. This is done by ensuring that any communication between the employees is done to the point - there is not much room for extra conversation.

1.3 Intended Audience and Document Overview

The intended user group of this document is the IT department of our client, as well as CS360's course staff.
The rest of this document starts with the requirements in the form that they have been communicated to us, followed by the formal logging of these requirements. The document concludes with model user stories and team information.
For all types of reader, please ensure that section 1 is well read.
If you are from ABL, sections 3 and 4 are more relevant to you, since section 2 is what you have communicated to us in its original form and so can be skipped.
If you are CS360's course staff, you should start with section 2 and continue with the appendices, section 3 and then section 4.

1.4 Definitions, Acronyms and Abbreviations

- ABL: Allied Bank Limited.
- Admin: Designated employees that have the administration role in the ILS system.
- Area: The broader category of a problem. For example: IT is the area of the Hardware category.
- Category: The category to which an issue belongs to. For example: Hardware is the category of all issues relevant to monitor or a computer mouse etcetera.
- Close issue: The final confirmation of resolution of the issue, done by the initiator, after the support role employee resolves the technical problem.
- Fix issue: When a support team member resolves an issue.
- ILS: Issue Logging System.
- Initiator: Any employee that logs in an issue on the app.
- Issue: Refers to a technical issue.
- Location: City of the branch where the issue was logged in.
- Log an issue: Fill a form with the required information about the technical issue and upload it.
- Manager: Manager of a support team.
- MOM: Minutes of Meeting
- RQ: Requirement.
- Sub-category: The more specific category of an issue. For example: any issues related to keyboard will have the sub-category as keyboard.
- Sub-location: Area of the branch where the issue was logged in.
- Support: Any employee part of a support team.
- Support Team: The employees at ABL that can resolve issues in a specific category form the support team of that category.
- Technical Issue: A hardware or software problem related to electronic devices that have been issued to an employee or a branch by the bank and is property of the bank.

1.5 References and Acknowledgments

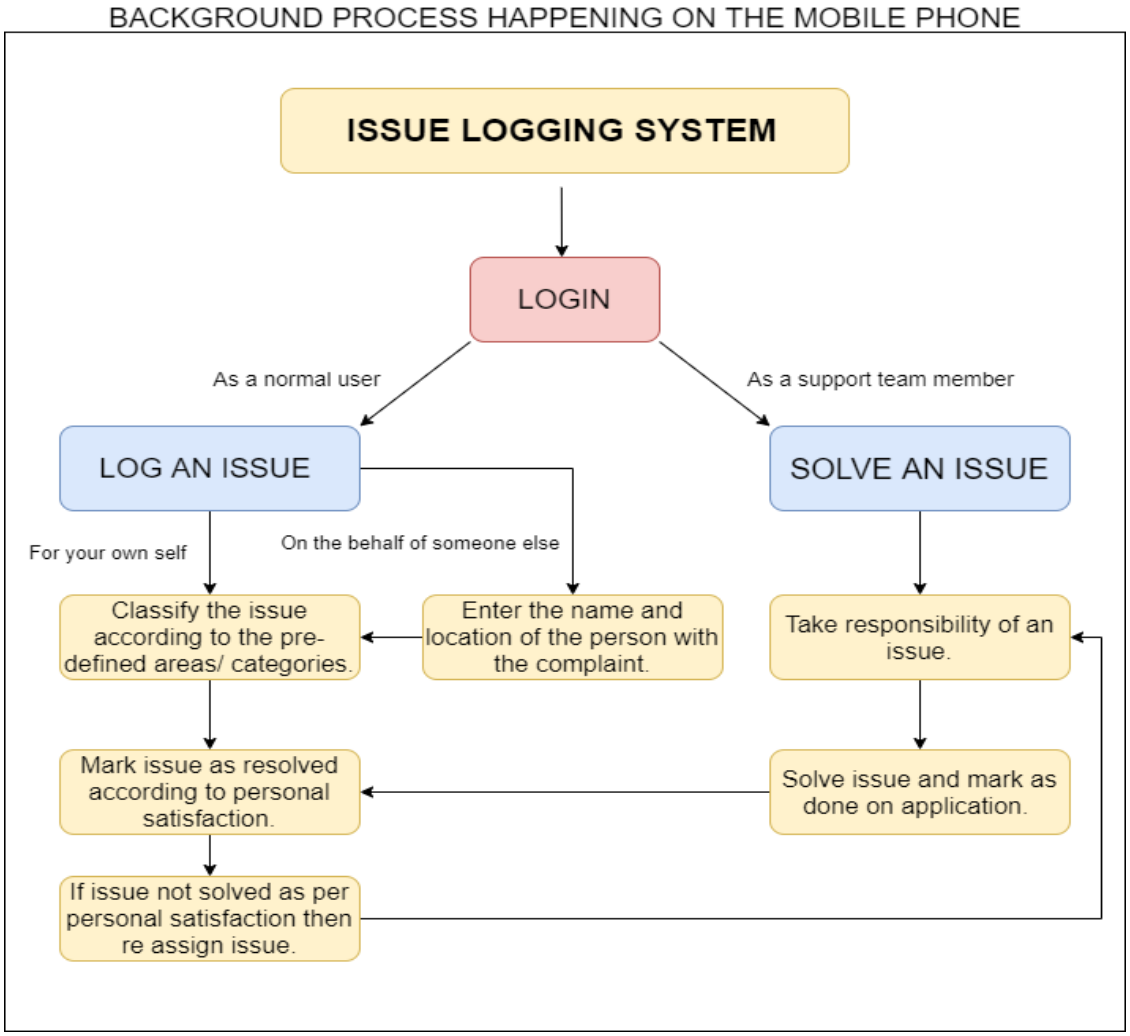
Geographic Locations of Allied Bank's branches in Pakistan: https://www.abl.com/business-banking/home-remittances/branch-and-atm-locator/

2 Overall Description

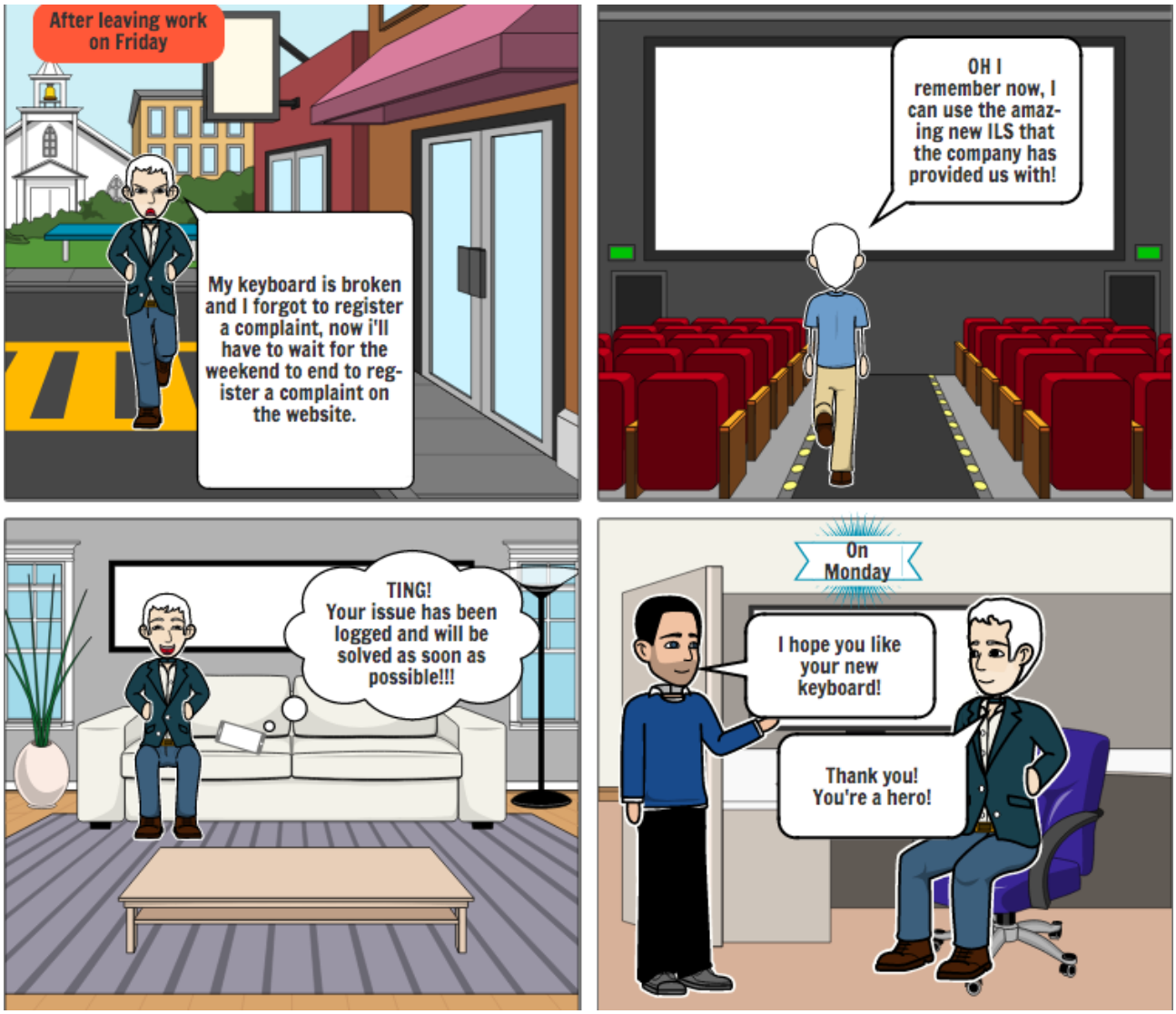
2.1 Product Perspective

The Issue Logging System being designed intends to be a prototype of the mobile application version of the Issue Logging System website already being used in Allied Bank Ltd. The product is being designed to log and view issues 'on the go' which will save time and prove to be a useful infrastructural asset for the company. The product is to be deployed into the Allied Bank Ltd server and software family by the company itself, if our prototype is accepted. A single pre-given login is to be used by the users and the software will be working using a dummy database, for the scope of this project, and a branch location list. Furthermore, the system will be a free open Issue Logging System which any company can integrate into their own system by simply pairing it up with their own employee database and making relevant changes.

Attached is a diagram to illustrate the major components of the overall system, subsystem interconnections, and external interface:



Attached is a diagram that illustrates the product interaction with the environment and the context of its use:



2.2 Product Functionality

I. The system will support one login with the help of an employee database which will have usernames and passwords stored already.

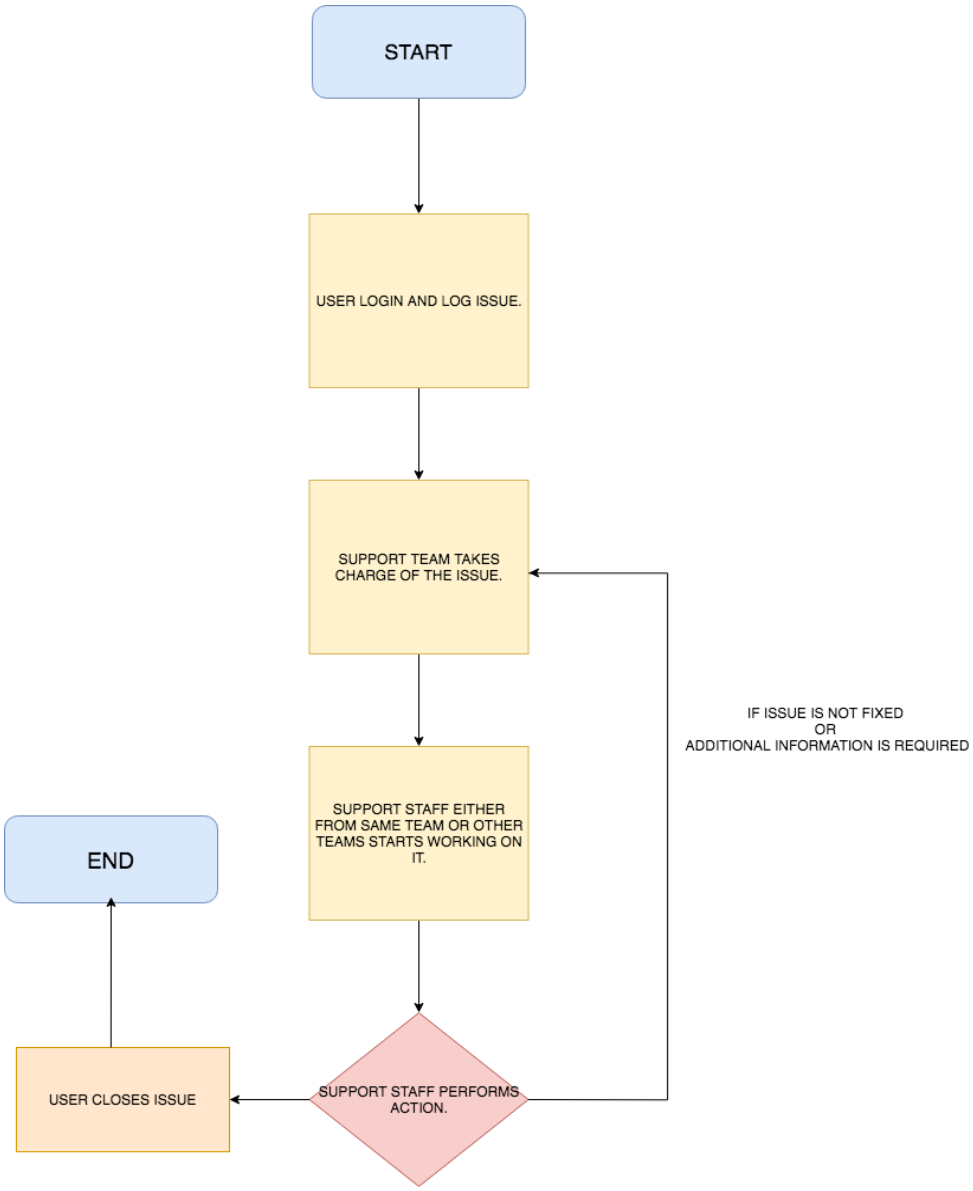
II. The user will login and the application will present functionalities with respect to the different roles already coded in the application according to their rank in the company.

III. The application provides options of category, sub-category, area, location and sub-location according to which the user will categorise the complaint/ issue being logged.

IV. As the issue logging will be completed there will be an issue logged into the panel where all the people who logged in as a support staff in the same city will have an option to take charge of any of the complaint/issue.

V. After resolving the issue, the support staff will mark the issue as fixed and only then will the initiator be able to mark it as closed, depending on the initiator's satisfaction with the resolution of the issue.

Attached is a Data Flow Diagram of the system:



2.3 Users and Characteristics

There are five different kinds of users that will be using this product namely, initiator, support, manager, monitor and admin, given in the order of importance from most important user to least. The options and rights these respective users have are specified below.

a. Initiator:

i. Any employee can login to the system

ii. Functions

1. Log support related Issues.
2. View status of own issues.
3. Mark status of own issues to closed or open it again.

b. Support:

i. User should be defined in System with Support role and be a part of defined Support Team

ii. Functions

1. Log support related issues.
2. View status of own issues.
3. Mark status of own issues to closed or open it again.
4. View issues of in-city branches matching their work category.
5. Assign open issues to themselves.
6. Mark resolved issues as fixed.
7. Reopen resolved issues.
8. Reassign issues to other team members.

c. Manager

i. User should be defined in the System with Manager role and part of defined Support Team.

ii. Functions

1. Log support related Issues.
2. View status of own issues.
3. Mark status of own issues to closed or open it again.
4. Assign open issues to themselves.
5. View Status of Issues pending with their team.
6. Mark resolved issues as fixed.
7. Refer Issue to other Support Team.
8. Assign/Re-assign Issue to other team members.
9. Monitor reports.

d. Monitoring.

i. Users should be defined in a System with Monitoring Role.

ii. Functions.

1. Log support related Issues.
2. View status of own issues.
3. Mark status of own issues to closed or open it again.
4. Monitor reports.

e. Admin

i. Users should be defined in System with Admin Role.

ii. Functions

1. Log support related Issues.
2. View status of own issues.

3. Mark status of own issues to closed or open it again.
4. Change Role of users.
5. Change Team of users.

2.4 Assumptions and Dependencies

1. The system assumes that all users who will have access to the application will be already included in the Allied Bank Ltd employee database and there will be no way to add or delete any user through this application.
2. The application totally depends on the Allied Bank database, i.e: if a user is not a part of the employee database then he/she cannot use the application.
3. For initial testing before deployment the system is dependent on a dummy server and database which will help in exhibiting the functionalities that will be implemented.
4. The system is only used within a company and will be used by the employees of the same company.
5. In the testing phase the scalability of the system will not be possible to be tested to the fullest as it will be next to impossible to replicate a large number of users as the company.
6. ABL has not been very open in sharing company related private information thus we have to make do with the information they have provided us.
7. ABL has denied us access to their main database and server which currently hosts the web ILS because to security reasons so we have to rely on our dummy database and server.

We are dependent on finding codes available online which will serve as the building blocks for our system.

3 Specific Requirements

3.1 Functional Requirements

Customer Functional Requirements

Login System

The system should ask every user to input their usernames and password.
Once the username and password has been entered, the system should compare them to the already provided credentials of the employee, as those stored in a dummy database.
Every login should successfully occur if the username and password entered matches the provided credentials of the employee, as stored in the dummy database.

Incorrect credentials should display an error message.

Roles

The system should be able to incorporate 5 different roles: Initiator, Support, Manager, Monitor and Admin.
Each role should be able to log in to the system that is altered to suit their role.
Every role should be able to log one's own support related issues through the ILS app.
Every role should be able to log other's support related issues through the ILS app on behalf of them. The system should be able to track original users by Name and staff ID.
Every role should be able to view the status of one's own issues through the ILS app.
Every role should be able to finally close a fixed request that they have logged or open it again.
Once logged in, a Support role user should be able to:

1. View issues of in-city branches matching their work category.
2. Assign open issues to themselves.
3. Mark resolved issues as fixed.
4. Reopen resolved issues.
5. Reassign issues to other team members.

through the app.

Once logged in, a Manager should be able to:

1. Assign open issues to themselves.
2. View Status of Issues pending with their team.
3. Mark resolved issues as fixed.
4. Refer Issue to other Support Team
5. Assign/Re-assign Issue to other team members.
6. Monitor reports

through the app.

Once logged in, a Monitor should be able to:

1. Monitor reports

through the app.

Once logged in, an Admin should be able to:

1. Change Role
2. Change Team

Emails and Notifications

The system should send out emails in the following scenario:

Sr #	Event	Email To	Email CC
1	Logging of New Issue	Initiator	Concerned Support Team
2	Issue Fixation	Initiator	
3	Issue Close	Initiator	Support team member who fixed Issue
4	Refer to Other team	Initiator	Respective Manager, Concerned Support Team
5	Refer to other team member	Initiator	Respective Manager, New resolver
6	Issue re-send to support team	Concerned support team members.	
7	User not accepted fixed issue	Concerned support team members.	

Push notifications should also be sent to both the ‘email to’ and ‘email cc’ columns as per the conditions of the above table.

General Functioning of the System

App should be able to add record to the database when issue is logged.

App should be able to update database especially when status of issue is changed.

App should be able to read database to retrieve issues logged.

App should be able to edit the role of any employee in the database.

App should be able to edit the team of any employee in the database.

App should be able to assign roles to users in the database.

App should allow logout when the logout option is selected.

App should allow fields for Location, Sub Location, Area, Category, Sub-category, Brief Description, Description, Options of logging on behalf on someone and the employee ID of the user who is facing the issue when logging an issue.

If time permits the developer, then the system should allow fields for attachments.

Automatic Functioning of the System

The system should auto close completed issues: on a daily basis, a service should be executed to close the issues automatically if they remained with fixed status for consecutive 2 days.

App should be able to fetch new updates to the logs and status of issues automatically.

The system should automatically escalate an Issue to the manager, if an Issue remains unattended or unresolved for 48 hours.

The system should automatically generate reports based on:

a) Total number of request against Area/ Category/ Sub-category/ Location

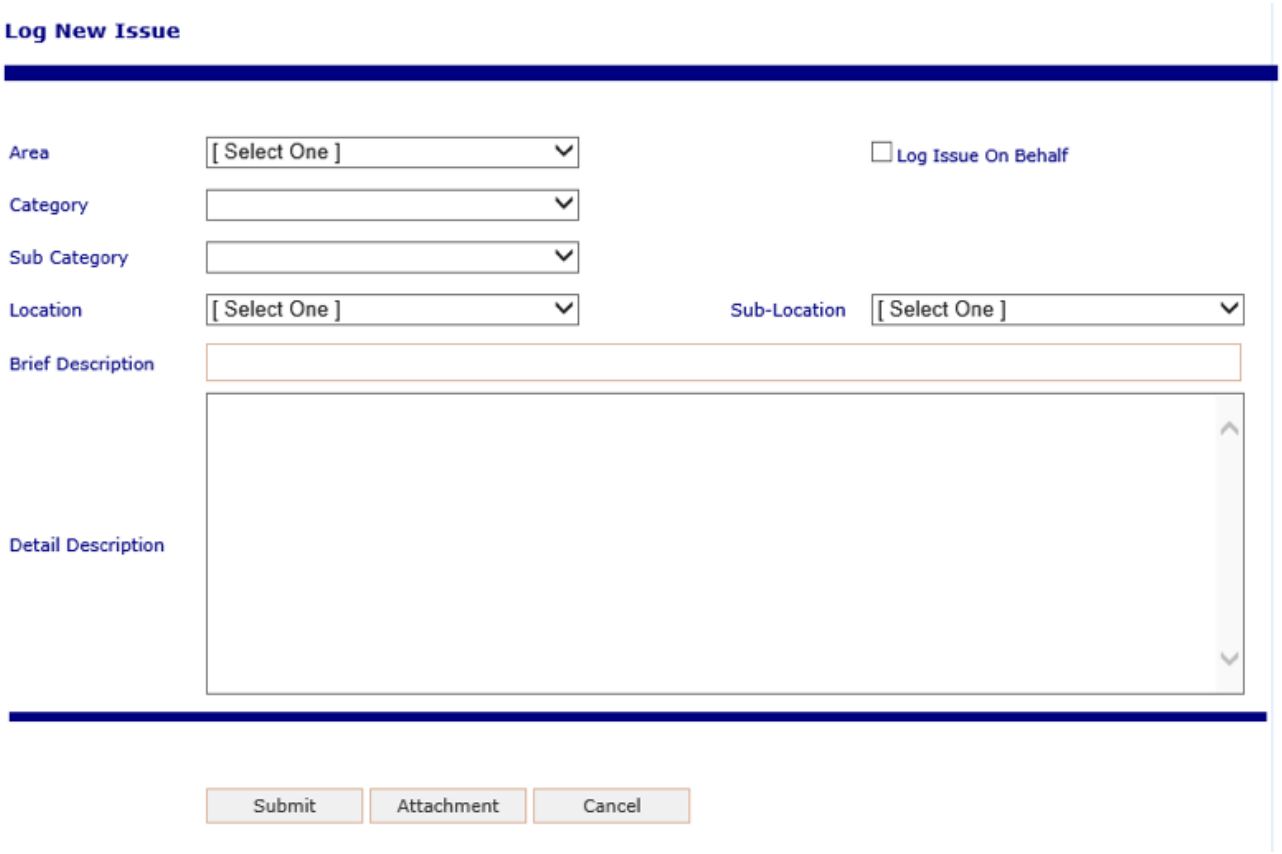
3.2 External Interface Requirements

3.2.1 User Interfaces

Since the ILS is a mobile application, it will be available on PlayStore and AppStore for downloading. It will be a touch mobile application. When opened, users will enter their credentials in the respective fields and then they will be logged into the portal according to their roles. We aim to make it a very interactive application which will keep the user informed about the progress of the issue which they log. For example, the user will receive push notifications when a support team member will take up the job, when the job is completed etc.

As far as accessibility features are concerned, we have planned to add some graphics/icons along the tabs to make it easier for the employees with low educational background, if any, to use. We inquired about the target audience of the application and we were informed that as far as this application is concerned, we are not required to add features for aiding disbaled people.

The issue logging form will have prepared lists like drop down menus to ensure that communication is to the point. For example, attached is a screenshot of ABL’s web ILS:



3.2.2 Hardware Interfaces

The application will have a very typical interface with hardware: the touch screen of the phone will be used as the primary source of input to the application and the outputs that the application will be generating will either be visible on the screen, or will be notified by push notifications, which will involve the speaker and the screen only, or by email. The supported device types are all mobile phones that are running on Android or IOS that can connect to the internet through Wi-Fi or cellular network.

3.2.3 Software Interfaces

As far as the software interfaces are concerned, once the employee selects an option from the menu of Location, Branch, Area, Category and Sub-Category the selected option will be communicated to the server which will then run pre-defined SQL queries to yield the relevant results. Apart from this once the user submits the issue, data from all the fields will be saved in a uniform template in the database. Moreover, after each action, the state of the issue will be updated, and the server will send the automated emails to the respective people from the email account stored in the database.

3.3 Use Case View

3.3.1 Use Case Table

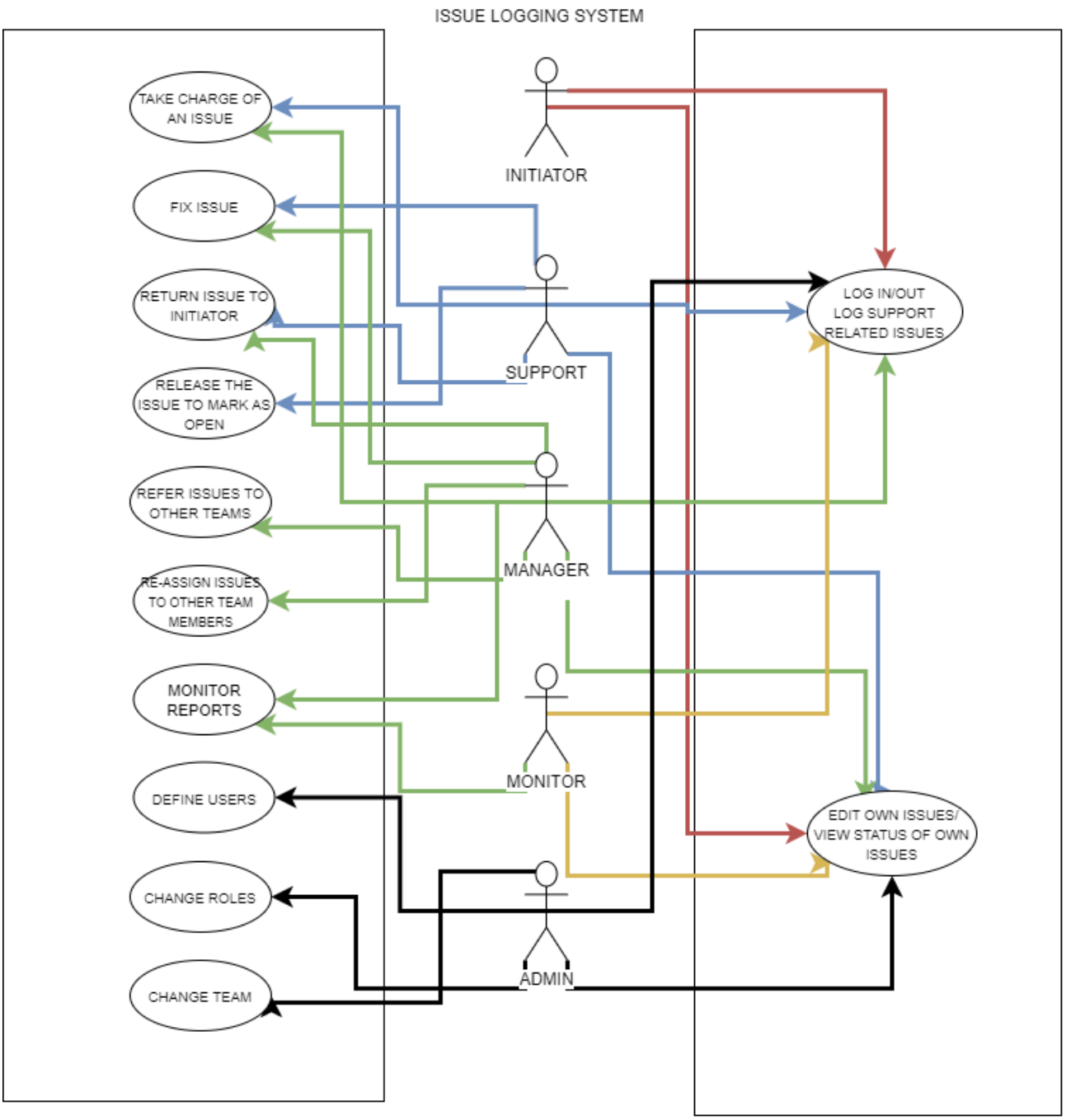
Use Case table

Primary Actor	Associated Use cases
Initiator	1. Login using credentials 2. Log support related issues 3. View status of own issue 4. Mark own issue as closed or open it again
Support	1. Login using credentials 2. Log support related issues 3. View status of own issue 4. Mark own issue as closed or open it again 5. View issues of in-city branches matching their work category. 6. Assign open issues to themselves. 7. Mark resolved issues as fixed. 8. Reopen resolved issues. 9. Reassign issues to other team members.
Manager	1. Login using credentials 2. Log support related issues 3. View status of own issue 4. Mark own issue as closed or open it again 5. Assign open issues to themselves. 6. View Status of Issues pending with their team. 7. Mark resolved issues as fixed. 8. Refer Issue to other Support Team 9. Assign/Re-assign Issue to other team members. 10. Monitor reports
Monitor	1. Login using credentials 2. Log support related issues 3. View status of own issue 4. Mark own issue as closed or open it again 5. Monitor reports

Admin	1. Login using credentials 2. Log support related issues 3. View status of own issue 4. Mark own issue as closed or open it again 5. Change Role 6. Change Team

3.3.2 Use Case Diagram

Attached is a picture of our use case diagram:


ISSUE LOGGING SYSTEM
TAKE CHARGE OF AN ISSUE
FIX ISSUE
RETURN ISSUE TO INITIATOR
RELEASE THE ISSUE TO MARK AS OPEN
REFER ISSUES TO OTHER TEAMS
RE-ASSIGN ISSUES TO OTHER TEAM MEMBERS
MONITOR REPORTS
DEFINE USERS
CHANGE ROLES
CHANGE TEAM
INITIATOR
SUPPORT
MANAGER
MONITOR
ADMIN
LOG IN/OUT LOG SUPPORT RELATED ISSUES
EDIT OWN ISSUES/ VIEW STATUS OF OWN ISSUES

3.3.3 Use Case Description

Use Case 1: Login to relevant window

Actors: initiator, support, manager, monitor, admin

Preconditions: An active connection to the internet network

Postconditions: Successful login to relevant window.

Basic Flow: The actor enters the credentials given to them by the bank that are already in the database. Upon successful authentication, the actor will be redirected to the role-specific window.

Alternative Flow: If the actor enters incorrect credentials, a login-error message appears.

Use Case 2: Issue Logging

Actors: initiator, support, manager, monitor, admin

Preconditions: The actor has authentic login credentials.

Postconditions: An issue complaint is registered in the system.

Basic Flow: The actor logs in and goes to the “Log new issue” window and selects the area, category and sub-category of the relevant issue and adds a brief description in the text box. The actor then clicks on the ‘submit’ button. Upon submitting, an email will be generated to the initiator and CC'd to the support team.

Use Case 3: View and update status of issue

Actors: initiator, support, manager, monitor, admin

Preconditions: An issue is logged in the system by the respective actor.

Postconditions: Knowledge of the issue (if it’s open or closed)

Basic Flow: The actor logs in and goes to the “View issue status” window where he locates his issue (if there exist multiple). The status of issue is visible next to the issue title. The status can be ‘open’: not resolved yet, ‘fixed’: resolved by the support team member, upon which an email will be generated to the users

specified in 3.1, or ‘closed’: meaning the issue has been resolved by the support team and has been closed by the initiator or auto-closed by the system after 2 days of ‘fixed’ status. If the actor is satisfied with the resolution of the issue, they can close the issue. After the issue is ‘closed’, an email will be sent to the initiator and the support team member who resolved it.

Alternative flow: The actor proceeds with the same steps in basic flow. The status is ‘fixed’ but if the user is not satisfied with the resolution, then the issue will be re-sent to the support team and an email will be sent to the support team.

Use Case 4: Assign Open Issues

Actors: Support Team, Manager

Preconditions: There exist open issues in the system.

Postconditions: The actor has assigned open issues to relevant personnel.

Basic Flow: The actor locates the list of issues and identifies the open ones. For both actors, they can assign issues to themselves or relevant support team members.

Use Case 5: Changing Roles

Actors: Admin

Preconditions: At least one employee exists in the system

Postconditions: The employee’s role has been changed

Basic Flow: The actor chooses an employee whose role has to be changed. The roles can be switched to any of the other roles that the employee isn’t currently.

4 Other Non-functional Requirements

4.1 Performance Requirements

- The application should be compatible with both android and IOS to ensure that it is accessible to all users.
- If the issue is not addressed within 48 hours, the issue should be escalated automatically with the manager to ensure timely resolution of the problem.
- The issue reported must be stored in the database and should be visible to the respected support team within 120 seconds after logging an issue.

- The issue resolved must be stored in the database and visible to the initiator within 120 seconds after issue is marked fixed by technician.
- The fixed issue must be notified to the initiator within 120 seconds after the issue is marked fixed by the support team.
- Any updates to the system should take no more than 120 seconds.
- Emails should be sent with a latency of no greater than 12 hours after each activity.

Time constraints have been applied to ensure consistency throughout all users using the application.

4.2 Safety and Security Requirements

4.2.1 Safety:

Use of dummy server:
The client has refused to authorize access to the server where their existing web ILS is running to assure safety and security of bank's server and other confidential details they can not afford to outsource. We are required to set up a dummy server for testing and running the application. Once the client approves the application, the Bank's IT department will themselves replace the dummy server with their own server.

Use of dummy database:
The client has refused to authorize access to the database where the Bank's existing web ILS is running, this is to assure safety and security of employees personal information. Client has however agreed to provide us with the structure of their existing database and using that structure we will design a dummy database which our application will use for running and testing, once client approves the application, the bank's IT department will themselves replace the dummy database and run the application with their own database.

Use of universal login by client:
To ensure that only the authorized personnel i.e. the bank's employees use the application and no other person gets authorized to login, the client wants the ILS application to login only via the employee's universal login credentials that every ABL Bank's employee is assigned for all logins. To cater this requirement, our application will not have a sign up option, only a sign in option will be given that will use only the employees universal login credentials stored in banks database for a successful sign in into the ILS application.

4.2.2 Security:

For the application, the highest possible level of security is required because the bank's data is highly confidential. A few possible security measures that we will ensure include:

Defense against XSS attacks:

We will assure that the application is not vulnerable to any attacks by XSS hackers. We will prevent this by using parameterized queries and defending our system from these attacks, by filtering out input where asked.

Prevention of creating or deleting users:

We will assure this by not allowing the application to add or delete records in the employee table, the admin can only edit the column of team or role, in the employee table, and the logs table.

4.3 Software Quality Attributes

4.3.1 Correctness:

The ILS application for ABL needs to be free from any error considering the users of the application. We aim to assure correctness because users are not expecting any errors. We aim to achieve that by multiple accuracy tests being conducted at 4 phases.

Phase 1: During development phase, after adding each feature to the application, the developer will assure that the desired output is achieved.

Phase 2: After completing the front-end development, our developers will test the application for any errors.

Phase 3: After application is complete, our entire team will try and test the application while catering any unexpected output issue that any team member observes.

Phase 4: A beta version will be provided to the client to test if users are facing any undesired output.

4.3.2 Portability:

The main reason why ABL wants an ILS mobile application is to assure the portability of this system because at this time they are using web ILS which restricts its users to the desktop. Therefore, keeping that in mind we aim to assure that the ILS mobile application also include complete features of the web ILS so users don’t feel the need to be present at their office desk to use a feature of ILS, they shall have access to all features of ILS system wherever and whenever they want.

4.3.3 Adaptability:

We aim to develop an application that performs well and gives the same user experience to users of all IOS and android platforms. We will assure this by using a

platform modifiable code base through flutter for the development phase of the application which means that the application can easily be adapted to adjust to any platform.

4.3.4 Usability:

Once the beta version of our application is ready, the users will be asked for feedback and improvements will be made if needed to assure best user friendly experience for the users. We ensure usability by using a GUI filled with graphics and colors to make our application more usable.

Appendix A – Top 10 User Stories

1. As an employee, I should be able to login with my universal credentials so that I do not have to make a separate account on ILS.
2. Each employee must be logged into the system according to their assigned roles so that they can only perform actions which they are authorized for.
3. As an initiator, I should be able to see drop down menus for selecting region, branch, issue category and issue sub category so that I can log the issue easily.
4. As a support team member, the issue should be visible on the application as soon as possible after it is logged in so that the issue can be taken up at the earliest.
5. As a support team member, I should be able to select a job so that I can assign it to myself.
6. As an initiator, as soon as the issue gets fixed by the support team, I should be prompted about the quality of work so that I can give feedback regarding the quality of work.
7. As an initiator, if I am not satisfied with the quality of the work, the issue should be raised again with the support team so that it can be fixed according to my expectations.
8. As a manager, if the job is not completed within 48 hours, I should be informed so that I can assign it to a support team and check why it wasn't taken up by any support team member.
9. As an employee, from issue logging to resolution, emails should be sent to the relevant people at each step so that the concerned people are kept in loop.
10. As an initiator, while logging an issue I should be able to add details so that the support team can easily understand my issue.

Appendix B – Architectural Spike (One Story)

Adding attachments while logging an issue falls under Architectural Spike.

The main reason for selecting this story is that we do not know the type or size of attachments which the users will be uploading, so we are aiming to cater most of the common types of uploads and hence do not know the complexity associated with storing attachments in database.

Appendix C - Group Log

Group meeting on 03/02/2020 with TA:

MOM: We discussed approaches to narrow down the requirements and how to go about deciding a tech stack for the project.

Group meeting on 12/02/2020 with TA:

MOM: We discussed languages and structural development (like the database, modes of users) of our system with Taimoor and also shared our collaborative work on researching the requirements of our system.

Group meeting on 15/02/2020:

MOM: We discussed the structure of SRS document and divided the work amongst all the five members of the group.

Group meeting on 19/02/2020 with TA:

MOM: We had a few questions about the SRS document and got the answers. Discussed a few queries about the language and environment too. Reported our progress on how we have divided the SRS document and we'll be compiling it by tomorrow.

Group meeting on 20/02/2020:

MOM: We met and shared our work. Our SRS document was in a very raw form. We sat together and read the document together and streamlined/improvised the document.

Group meeting on 22/02/2020:

MOM: We further improvised the SRS document.

Group meeting on 24/02/2020 with TA:

MOM: We had some minor confusions regarding the SRS document. Discussed them with TA and sorted them out.

Group meeting on 24/02/2020:

MOM: We worked on producing the final draft of the SRS document.

Appendix D – Contribution Statement

Name	*Contributions in this phase*	*Approx. Number of hours*	*Remarks*
Aadam Nadeem	Introduction,Specific Requirements and Compilation	15	Outstanding
Maleeha Masood	Specific-Requirements, Introduction and Compilation	15	Outstanding
Malik Ali Hussain	Appendices,Other Non-functional Requirements and Compilation	15	Outstanding
Muhammad Raheem Zafar	Overall Description, Specific Requirements and Compilation	15	Outstanding
Shahrukh Kemall	Other Non-functional Requirements, Appendices and Compilation	15	Outstanding