

**Woldia University**

**Institute of technology**

**School of computing**

**Final year industrial project submitted in fulfillment of the requirements of bsc degree in information technology**

**Project title: Android based woldia university maintenance request tracking system**

**By**

|  |
| --- |
| Dawit G/hiwot WDU1200619 |
| Habtamu Abebaw WDU111917 |
| Zewdu Tafese WDU1202268 |
| Helari Seyifu WDU1204641 |
| Birhane Getachew WDU1200498 |

Main Advisor: Mr. Solomon A.Msc

Co-advisior: Chilote D.Msc

**7/3/2023 G.C**

**Woldia, Ethiopia**

**This is approved by examiner committee that we have read this project and that in our options it is fully adequate in scope and quality, as a project for the degree of Bachelor of Science.**

**Name of Advisor Date Signature**

**1.**

**2.**

**Name of Examiner Date Signature**

**1.**

**2.**

**3.**

**4.**

# **Abstract**

Maintenance request tracking system is an appropriate technology that aims to solve maintenance need problems in woldia university facility by adapting ICT to current technologies used in woldia university. The maintenance request tracking system in woldia university has been around for long time. Currently in woldia university, most maintenance request tracking system is only possible with the help of Papers or orally. because of this the delivery of the request papers from one facility to another facility member might take a while: so, this system intends to computerize existing manual maintenance request tracking system. After major problems of the existing system in maintenance request tracking system are reviewed, a clear list of requirements was formulated using software Requirement specification and designed using standard UML (unified modeling language) tools. Project is implemented by object-oriented paradigm with MySQL, java (Android), PHP. In this system the Request sender requests any issues regarding maintenance; Any Facility after logging in to the maintenance request tracking system can view an incoming message of maintenance request, approve request, forward request, and assign maintenance technicians for the request. The project is significant on saving time and speed, decrease man power wasted. The project is efficient and effective on delivering quality information and cost saving

# Table of Contents

[**Abstract** i](#_Toc129093473)

[Table of Contents ii](#_Toc129093474)

[**List of tables** iv](#_Toc129093475)

[**List of figures** v](#_Toc129093476)

[**Acronyms** vi](#_Toc129093477)

[**Chapter One: Introduction** 1](#_Toc129093478)

[1.1 Introduction 1](#_Toc129093479)

[1.2 Background of the organization 1](#_Toc129093480)

[1.3 Statements of the problem 2](#_Toc129093481)

[1.4 Project objective 2](#_Toc129093482)

[**1.4.1** **General objective** 2](#_Toc129093483)

[**1.4.2** **Specific objective** 2](#_Toc129093484)

[1.5 Scope of the project 3](#_Toc129093485)

[**1.5.1** **Scope of the project** 3](#_Toc129093486)

[1.6 Methodology 3](#_Toc129093487)

[**1.6.1** **Requirement gathering methods** 3](#_Toc129093488)

[**1.6.2** **System analysis and design Methodology** 4](#_Toc129093489)

[**1.6.3** **Software development model** 4](#_Toc129093490)

[**1.6.4** **Implementation Methodology** 5](#_Toc129093491)

[1.7 Feasibility of the project 6](#_Toc129093492)

[**1.7.1** **Economic Feasibility** 6](#_Toc129093493)

[**1.7.2** **Tangible Benefit.** 6](#_Toc129093494)

[**1.7.3** **Intangible Benefit** 6](#_Toc129093495)

[**1.7.4** **Technical Feasibility** 6](#_Toc129093496)

[**1.7.5** **Operational Feasibilities** 6](#_Toc129093497)

[**1.7.6** **Behavioral/Political feasibility** 6](#_Toc129093498)

[1.8 Beneficiaries or significant of the project 7](#_Toc129093499)

[**Chapter Two: Business area analysis and requirement definition** 8](#_Toc129093500)

[2.1 The Existing System 8](#_Toc129093501)

[**2.1.1** **Drawbacks of the existing system** 8](#_Toc129093502)

[**2.1.2** **Business Rules in the existing system** 8](#_Toc129093503)

[2.2 Purposed system 9](#_Toc129093504)

[**2.2.1** **Business rule of proposed system documentation** 9](#_Toc129093505)

[2.3 Requirement Analysis 10](#_Toc129093506)

[**2.3.1** **Functional requirement** 10](#_Toc129093507)

[**2.3.2** **Nonfunctional Requirements** 11](#_Toc129093508)

[2.4 System Requirement 11](#_Toc129093509)

[**2.4.1** Hardware requirements: 11](#_Toc129093510)

[**2.4.2** Software requirements: 11](#_Toc129093511)

[**2.4.3** System Use case 11](#_Toc129093512)

[**2.4.2 Use case identification and description** 13](#_Toc129093513)

[2.4.2 State chart diagram 24](#_Toc129093514)

[2.4.3 Activity Diagram 25](#_Toc129093515)

[2.4.4 Sequence diagram 28](#_Toc129093516)

[2.4.5 Collaboration diagram 32](#_Toc129093517)

[2.4.6 Conceptual modeling: Class diagram 34](#_Toc129093518)

[References 35](#_Toc129093519)

[Appendix 36](#_Toc129093520)

# **List of tables**

[Table 1 Login 14](#_Toc129032404)

[Table 2 Manage account 15](#_Toc129032405)

[Table 3 Request maintenance needs 16](#_Toc129032406)

[Table 4 View request 17](#_Toc129032407)

[Table 5 Report activity 18](#_Toc129032408)

[Table 6 View report activity 19](#_Toc129032409)

[Table 7 Approve/disapprove user 20](#_Toc129032410)

[Table 8 Assign maintenance technician 21](#_Toc129032411)

[Table 9 Track maintenance request 22](#_Toc129032412)

[Table 10 Reject request 23](#_Toc129032413)

# **List of figures**

[Figure 1 Use case diagram 12](#_Toc129067594)

[Figure 2 state chart for view request 24](#_Toc129067595)

[Figure 3 State chart for maintenance request 24](#_Toc129067596)

[Figure 4 State chart for login 25](#_Toc129067597)

[Figure 5 State chart for manage user accounts 25](#_Toc129067598)

[Figure 6 Activity diagram for request sender registration 26](#_Toc129067599)

[Figure 7 Activity diagram for request maintenance 27](#_Toc129067600)

[Figure 8 Activity diagram for login 27](#_Toc129067601)

[Figure 9 Activity diagram for Approve Login request 27](#_Toc129067602)

[Figure 10 Activity diagram for send maintenance report 28](#_Toc129067603)

[Figure 11 Sequence diagram for login 29](#_Toc129067604)

[Figure 12 Sequence diagram for view request 29](#_Toc129067605)

[Figure 13 Sequence diagram for request maintenance needs 30](#_Toc129067606)

[Figure 14 Sequence diagram for report maintenance activities 30](#_Toc129067607)

[Figure 15 Sequence diagram for reject request 31](#_Toc129067608)

[Figure 16 Sequence diagram for Track maintenance request 31](#_Toc129067609)

[Figure 17 Collaboration diagram of login 32](#_Toc129067610)

[Figure 18 Collaboration diagram of create account 33](#_Toc129067611)

[Figure 19 Collaboration diagram of view request 33](#_Toc129067612)

[Figure 20 Class diagram 34](#_Toc129067613)

# **Acronyms**

|  |  |
| --- | --- |
| **Acronyms** | **Definitions** |
| WUMRTS | Woldia university maintenance request tracking system. |
| MYSQL 5.6 | Structured Query Language. |
| XAMPP 2.3 | Cross-Platform (X), Apache (A), MySQL (M), PHP (P) and Perl (P). |
| RAD | Requirement analysis document. |
| WU | Woldia University. |
| PHP | Hypertext preprocessor |
| OOSAD | Object oriented system analysis and design |
| UML | Unified modeling language |
| CD | Compact disc |
| SDLC | System design life cycle |

# **Chapter One: Introduction**

## Introduction

Woldia university activities require intensive fixed asset use, fixed assets often need maintenance and unmaintained create business operation discontinuities. The maintenance request tracking system allows a user to submit maintenance requests on a variety of issues.

Current maintenance is poorly performing because the time from request to actual maintenance takes too much time and is communication intensive, maintenance request tracking system provides information for the maintainers and facility, eliminate paper work orders, never miss a request, and get things done faster. Maintenance request tracking system is an automated system that used for facility to give maintenance services for all woldia university community.

## Background of the organization

As we know that, in Ethiopia there are more than eighty nations and nationalities, the sensation and nationalities have their own culture, norms, beliefs, eating and language. University is a special kind of community, in recent years. Ethiopian University is becoming the first places where students from different cultural background meet together. woldia University, which is found in Amhara Region, is one of Ethiopian university. In this university there are students who come together from every corner of the country with different cultural background. As a result, they face problem in communication, so, it needs some scientific approaches to full fill this gap among students [1].woldia university was established through the council of ministers Regulation No223/2011 issued on 2004. it is found in North wollo, Amhara region, Ethiopia. In the Faculty of Technology, Faculty of Natural and Computational Science, Faculty of Educational and Behavioral Science, College of health Science, College of Agriculture, Faculty of Social Science and Humanities, School of Land Administration, Faculty of Business and Economics woldia university has three campuses, namely, the main campus called woldia university, mersa campus of Agriculture 30 km far from main campus and Lalibela campus of Horticulture and Tourism 169 km far from main campus [2].

## Statements of the problem

Currently, maintenance tasks at woldia university are performed manually, which has the following drawbacks: management is challenging, time and other resources are wasted, to name a few. manually keeping track of maintenance requests wastes materials like paper, photocopies, etc. As there is no automated maintenance work communication on campus, it is challenging to follow up on operations in a timely manner. The current manual system, which contains all of the activities, has the following issues: Insecurity exists due to factors including fire, theft, and the complexity of data storage. Many steps are used to physically perform maintenance. The provision of maintenance services that users require is laborious and takes a lot of time. It takes a lot of time and effort to obtain the essential report on the resources that are being maintained. Order delivery was delayed as a result of manual interaction between users. the wasteful and lavish use of resources. The loss of the maintenance files was caused by their paper storage. Tracking the request to the facility uses a lot of resources. manual record keeping that is not dependable. Requesting manual maintenance causes controversy and confusion. Information about preserved things is difficult to obtain. Data storage and retrieval are challenging tasks. The files weren't correctly stored. The number of maintainers and the responsibilities assigned to them are unknown to the facility. Slow maintenance results in low-quality educational services.

## Project objective

### **General objective**

The general objective of this project is to develop android based maintenance request tracking system for woldia university facility management.

### **Specific objective**

In Order to achieve the general objective, the following specific objectives are needed.

* To gathering the current manual system.
* To analyze functional and nonfunctional requirements
* To design easy user interface and normalized database model
* To implement the designed system
* To test the implemented system.
* To maintain if there defects.

## Scope of the project

### **Scope of the project**

The project emphasis on automating information of the maintenance request tracking system in woldia university. That is to change the maintenance request tracking manual system into automated system. The project attempt to design a system that could be applicable to the woldia universities facility management to gain every user maintenance request simply. This maintenance request tracking system has the following operational domains:

* The system is android based application and used for maintenance service.
* Provides basic information of the request to facility manager.
* Receive notification when new request come
* Sending maintenance request.
* Sending maintenance activity report.
* Login privilege is given for registered user only
* To create account for users, the privilege is given for administrator only.
* The system track maintenance request from users to the facility.
* The system responds for the request of management.
* The system provides basic information of the request.
* The system will operate in any networked environment.

## Methodology

### **Requirement gathering methods**

Data collection methods are the most important part of our project to find the main requirements of system and how to understand the system is does. To gather the information, we used Primary data source collection methods that mentioned as follow: -

#### **Interview**

The main data source for this project was the woldia university facility office. In order to get crucial information, we need for the project we would like to have conversation with woldia university facility management. Initially there is a continuous discussion with the facility to get detail information about the maintenance request. To see how the current system works, the problem associated with the current system, skill that is needed by the maintenance request and to reduce the problem that they are facing at present. The project is to be carried out by a team of students.

#### **Observation**

The project is to be carried out by a team of five students. To understand system process we collect information by physical observation existing system.

#### **Document Analysis**

We used this method as one means of data collection because it is relatively inexpensive good source of background information, unobtrusive, provides a behind-the-scenes look at a program that may not be directly observable, and may bring up issues not noted by other means. This method of collecting a data is analyzing the document prepared in the existing system. The team analyzed different documents like forms and report samples in the existing system. In form samples we used information like maintenance request forms, requester forms so on, and from the report samples the team gathered information about the day-to-day activities of the institution.

### **System analysis and design Methodology**

Here for the analysis of our project we design method specifically UML (Unified Modeling Language) model. We have selected this because of the following advantages: -

* To simplify the design and implementation of complex programmers.
* To make it easier for teams of designers and programmers to work on a single software project.
* To decrease the cost of software maintenance.
* Increase reusability.
* Increased consistency among analysis, design and programming activities.
* Improved communication among users, analysis, design and programming.

Object oriented system analysis and design methodology: Is software development methodology by building self-contained modules or objects. This methodology has the following features increased reusability, increased extensibility, provide quality, and reduced maintenance burden and managed complexity due to this we select OOSAD methodology.

### **Software development model**

Iterative process starts with a simple implementation of a subset of the software requirements and iteratively enhances the evolving versions until the full system is implemented. At each iteration, design modifications are made and new functional capabilities are added.

The software development model that we have used is iterative model among other models because of it enables us to go forward and backward as it is necessary. For example, if the university will support us, we can return back and advance the system iteratively. In this Model, you can start with some of the software specifications and develop the first version of the software. After the first version if there is a need to change the software, then a new version of the software is created with a new iteration. Every release of the iterative model finishes in an exact and fixed period that is called iteration.

The iterative model allows the accessing earlier phases, in which the variations made respectively. The final output of the project renewed at the end of the Software Development Life Cycle (SDLC) process [3].

### **Implementation Methodology**

There are different types of software development methodologies from this we used: -

* We use java as scripting language.

Hardware tools: -

* Flash disk (8GB): - required for data movement to store & transfer data from one PC to another.
* Paper and pen: for writing all necessary documentations associated with the project.
* Laptop and desktop: - almost all tasks of our project are performed on the computer.
* Disks (CD, DVD): necessary for the movement, backup and recovery mechanism of relevant data.

Software tools: -

* MS word 2016: for writing documentation.
* Microsoft PowerPoint 2016: -Is software that we use for presentation purpose.
* MYSQL 5.6 database: for Data base server purpose.
* Xampp 2.3 Server: - to test the system.
* Wonder share Edraw max version 8.0: - for diagrams like use case, sequence diagrams, class diagrams, deployment diagrams.
* Android studio: since our system is Android based, it is very necessary required.

## Feasibility of the project

The feasibility study is used to investigate the proposed system in multiple dimensions. It is used to indicate whether the system is feasible or not [4].

### **Economic Feasibility**

Economic feasibility is the process of identifying the financial benefits and costs associated with the project being developed. So, the project is economically feasible because the project reduces the cost of the resources. But economic feasibility is expressed as cost- benefit analysis. Costs- our system uses new technology and have centralized database cannot need more resources. It requires a minimum amount of cost.

### **Tangible Benefit.**

Tangible benefits: are benefits derived from the creation of an information system that can be measured in beer and with consistency, it reduces resource used and time spent during manual one such as: -

* + - * + Stationary cost
        + Hardware cost
        + Software cost

### **Intangible Benefit**

* Increased speed of activity.
* Increased flexibility in the system.
* Error reduction.

### **Technical Feasibility**

The proposed system can be easily maintained and repaired without requiring high experts or technical supports, because the system will be installed in adaptable technology’s. So, the system is technically feasible. Our project is technically feasible. Because, the project is practically applicable within the current technology.

### **Operational Feasibilities**

The system performs all operations to achieve the specified objective, User friendly and interactive with the environment and the system will perform all operations that the organization runs. And it will not have any difficulty or procedures to perform the operation of the system.

### **Behavioral/Political feasibility**

The system to be developed is not conflict with any government directives, because it gives services for the request sender2 effectively and efficiently, all the stakeholders also agreed before the system developed. So, the government is profitable and the system will be politically feasible.

## Beneficiaries or significant of the project

Upon the completion of the project, the following are the main beneficiary users:

* The project enables woldia university facility management to work its plan effectively and efficiently and saves their manpower, resources (papers, pencils etc.) which loses for tracking maintenance request.
* woldia university cafeterias, dormitories, class rooms, department staffs, computer laboratory, libraries request maintenance easily in short period of time.
* Woldia university student union request to facility at any time.
* shall enable efficient maintenance service to all users

# **Chapter Two: Business area analysis and requirement definition**

## The Existing System

Currently the maintenance issues in woldia university have no automated system for requesting the maintenance issues and keeping maintenance data. They request maintenance manually and information is processed by papers this is not secured and wastes the resource. The maintenance process of existing system is briefly described below.

The request sender writes things to be maintained and take request manually to the facility. Then facility approves, accept or reject the request by viewing the maintenance request. If they accept the maintenance request, they order the maintenance technicians. After the maintenance is done the maintenance technicians report the maintenance activity to the facility manually.as well as the maintenance around staffs, libraries, class rooms, cafeterias are collected by staff heads, library head, class room head and cafeterias head respectively, take to facility manager and maintenance is done by same process.

### **Drawbacks of the existing system**

* The existing system is time consuming.
* The existing system uses paper work. And maintenance request papers can be lost

### **Business Rules in the existing system**

A business rule is effectively an operating principle or polices that must be fulfilled and obligated in order the system will function properly and effectively. It often pertains to access control issues, business calculations, or operating polices and principles of the organization [5].

* **BR1:** Authorize to the system maintenance’s manager and facility manager, request sender must have a valid university id card.
* **BR2:** In order to get maintenance services after request the request sender must be inside the campus.
* **BR3:** Maintenance request can be submitted using paper or orally.
* **BR4:** Facility manager should not assign maintenance technician in more than one request task.
* **BR5:** The department head needs to write maintenance approval paper before requesting maintenance need.

## Purposed system

Maintenance service is used on return on investment and resource utilization of assets. The maintenance tracking system has many stake holders such as Admin, facility manager, request sender, maintenance technician etc.

Maintenance tracking system involves activities such as reporting the assets of building that needs repairs like broken lockers, chairs, keys, windows mirrors, water pump etc. In addition to this it has the activities of reporting maintenance needs of general service, computer maintenance, liquid waste, beatification of campus, solid waste, cleaner outsource etc. The purpose of the proposed system is to automate the maintenance issues in the university such as maintenance in the student’s dormitory, staffs, ICT, cafeterias, libraries etc. and all maintenance activities that are done in the campus.

The purpose of maintenance request tracking system project is to speed up the maintenance activities of buildings of library, dormitory, computer laboratory, cafeteria, staffs, class rooms, etc. and other equipment’s, identifying the areas needing repairs or improvements and assigning personnel and resources to make the repairs.

### **Business rule of proposed system documentation**

In order to effectively operating the software, satisfy a principles or policy with regarding to access control issues, and principles of the organization. Therefore, the system operates this software are identified by the following business rules.

**BR1:** Authorize to the system maintenance’s manager and facility manager, request sender must have a valid university id and password.

**BR2:** To request form maintenance service request sender must have approved and logged in.

**BR3**: In order to get maintenance services after request the request sender must be inside the campus.

**BR4:** request sender had right to request for any maintenance request need.

**BR5:** The maintenance request should be assigned by facility manager.

**BR6:** New maintenance request have to received and organized by facility manager.

## Requirement Analysis

### **Functional requirement**

Functional requirements are requirements that are used to capture the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform and a description of activities and services a system must provide. This requirement describes the interaction between the system and its environments independent of its implementation. The environment includes the user and any other external system with which the system interacts. It expresses a statement of exactly what the system must do.

* + - * + The system should enable the request sender to request maintenance needs and reporting the maintained things.
        + The system allows to the request sender to view request of its on,
        + Enable the request sender to cancel request of its on.
        + Enable the request sender to view maintenance schedule date.

The system should enable facility manager head to assign requests

Enable facility manager to view maintenance activity.

* + - * Enable facility manager to viewing complain.
      * The system should grant facility manager to view reports.
      * The system allows facility manager to view requests.
        + The system should enable the maintenance technician to report maintenance activity.

The system should grant the maintenance technician to generate report.

* + - * Enable the Admin to view list of users

The system should enable the admin to manage user accounts.

The system allows to the admin to change password.

* + - * The system allows to the admin to deactivate user accounts.
      * Enable the admin to backup.

### **Nonfunctional Requirements**

Nonfunctional requirements describe user-visible aspects of the system that are not directly related with the functional behavior of the system. These requirements do not directly affect the performance of the system but are nevertheless important. Nonfunctional requirements include a broad variety of requirements that apply to many different aspects of the system, from usability to performance. The system shall protect unauthorized access to system services and underlying data. The system provides an access to privilege to an authorized user by giving account for each and every special function. The system gives the privilege for user to change password.

**The Non-functional requirement of our proposed system includes:**

**Availability:** the system should be available at any time, if by any means the system fails, the system should store a backup database.

**Performance:** the system should be responsibly fast in order to access the required crime information’s.

**Security:** The system must confidentially be controlled from being accessed by unauthorized users. Virus scanning software should be installed to protect from viruses. Duplicate record must be discarded by the administrator.

**Maintenance:** New additional features can be added if required and also the system components i.e., memory, disk, drives shall be easily serviceable without much alteration in the code.

* 1. System Requirement

### Hardware requirements:

Android phone or tablet computers having typical storage capacity and processing speed and for another server, enough amount of Space, Desktop computer.

### Software requirements:

Operating System: android

Setup: - java (Android studio 2021.3.1), MySQL 5.6, EDRAW MAX 8.0 , android xml (user interface),Microsoft office word 2016.

### System Use case

#### Use case Diagram

A use case diagram is one of the Unified modeling languages that indicates an interaction between users and a system. It captures the goal of the users and the responsibility the system to its users. It is the functionality of the system or the service provided by the new system. The main purpose of a use case diagram is to show what system functions are performed for which actor. These diagrams contain the following elements:

* **Admin**: is the oversees team members in a certain department to ensure its performing effectively.
* **Facility manager**: Manages users maintenance requests.
* **Maintenance technician:**  It is the person who perform maintenance tasks
* **Request sender:** It is the person who sends maintenance needs   
  in order to get services



Figure 1 Use case diagram

### **2.4.2 Use case identification and description**

In the Use Case describes the functionality to be built in the proposed system, which can include another Use Case's functionality or extend another Use Case with its own behaviour. The most important and basic use cases of this system are the following:

**List of use cases**:

* **Use Case ID** 1: Login
* **Use Case ID** 2: Manage account
* **Use Case ID** 3: Request maintenance needs
* **Use Case ID** 4: View request
* **Use Case ID** 5: Report maintenance activity
* **Use Case ID** 6: View report activity
* **Use Case ID** 7: Approve/disapprove user
* **Use Case ID** 8: Assign technician
* **Use Case ID** 9: Track maintenance request
* **Use Case ID** 10: Reject request

Use case diagram for WUMRTS.

#### Use case description

Use Case Descriptions: A diagram showing your use cases and actors may be a nice starting point, but it does not provide enough detail for your system designers to actually understand exactly how the system's concerns will be met. How can a system designer understand who the most important actor is from the use case notation alone? What steps are involved in the use case? The best way to express this important information is in the form of a text-based description every use case should be accompanied by one.

Table 1 Login

|  |  |  |
| --- | --- | --- |
| Use case id | **Use Case ID** 1 | |
| Use case name | Login | |
| Actors | Maintenance technician, Request sender, Admin, Facility manager | |
| Description | The authentication for authorized users in the system. | |
| Precondition | Any user must have university id and password. | |
| Basic flow of action | Actor action | System response |
| **Step2:** university id and password | **Step1:** system show screen  **Step3:** the system checks the authentication of user name and password  **Step4:** system display user page |
| Post condition | System transfer control to user main screen to precede actions. | |
| Alternative action | **A.** The university id and password is invalid.  **1.** The system displays error message.  **2.** The system continues at step 2 to fill university id and password again. | |

Table 2 Manage account

|  |  |  |
| --- | --- | --- |
| Use case id | **Use Case ID** 2 | |
| Use case name | manage account | |
| Actor | Admin | |
| Precondition | The Admin must login to control the account | |
| Description | This activity is performed when the Admin want to manage the account | |
| Basic course of action | Actor action | System response |
| **Step1:** Admin enter user name and password  **Step4:** admin select mange ac  count page: -  1.create account  2. View list of users  3. Activate/deactivate account  **Step6** Admin enter user account information | **Step2:** the system checks the authentication of user name and password  **Step3:** the system display Admin page  **Step5:** System display create account page.  **Step7:** system check creates user account information  **Step8:** System creates user account |
| Post condition | The system Admin successfully creates, Activate/deactivate account | |
| Alternative course of action | 1. Invalid information entry. 2. The system displays error message.   **2.** Go to **step 6** to fill again. | |

Table 3 Request maintenance needs

|  |  |  |
| --- | --- | --- |
| Use case id | **Use Case ID** 3 | |
| Use case name | Request maintenance needs | |
| Actor | Request Sender | |
| Description | Request sender sends maintenance request needs | |
| Precondition | The Request sender must login to add new information | |
| Basic course of action | Actor action | System response |
| **Step1:** Request sender enter user name and password  **Step4:** Request sender fill the request form.  **Step5:** Request sender send the request | **Step2:** the system checks the authentication of university id and password  **Step3:** the system displays Request sender page  **Step6:** The system gives conformation of sent. |
| Post condition | The system Requested information will be viewed by an authorized user | |
| Alternative course of action | 1. If the submitted new request is not valid. 2. The system displays error message. 3. Go to **Step4** to submit the add again. | |

Table 4 View request

|  |  |  |
| --- | --- | --- |
| Use case id | **Use Case ID** 4 | |
| **Use case name** | View request | |
| Primary actors | **Facility manager** | |
| Description | Facility manager can see the request that are submitted from the request sender. | |
| Precondition | Go to the site and register | |
| Basic flow of action | **Actor action** | **System response** |
| **Step1:**  **Facility manager enter user name and password**    **Step4:** **Facility manager select view request.**  **Step6:** **Facility manager view request.** | **Step2:** the system checks the authentication of university id and password  **Step3:** system display Facility manager page  **Step5:** system display request records |
| Post condition | Facility manager views the submitted requests. | |
| Alternative action | **1.** The system displays error message.  **2.** Go to **step4** to view comment again. | |

Table 5 Report activity

|  |  |  |
| --- | --- | --- |
| Use case id | **Use Case ID** 5 | |
| Use case name | Report activity | |
| Primary actors | Maintenance technician | |
| Description | **Report finished maintenance task** | |
| Precondition | The Maintenance technician must be log in the system | |
| Basic flow of action | Actor action | System response |
| **Step1: Maintenance technician enter university id and password**  **Step4:** **Maintenance technician select report activity**  **Step6: Maintenance technician fill the form and send** | **Step2:** the system checks the authentication of university id and password  **Step3:** system display Maintenance technician page  **Step5**: the system display report activity page  **Step7: The system gives conformation of sent.** |
| Post condition | The **Maintenance technician** send report | |
| Alternative action | 1. If the submitted new request is not valid. 2. The system displays error message. 3. Go to **Step6** to submit the add again. | |

Table 6 View report activity

|  |  |  |
| --- | --- | --- |
| Use case id | **Use Case ID 6** | |
| Use case name | View report activity | |
| Primary actors | Facility manager | |
| Description | **Facility manger want to see report of maintenance activity from maintenance technician and request sender** | |
| Precondition | The Facility manager must be log in the system | |
| Basic flow of action | Actor action | System response |
| **Step1:** **Facility manager enter university id and password**  **Step4:** **Facility manager select view report**  **Step6: Facility manager insert report date** | **Step2:** the system checks the authentication of university id and password  **Step3:** system display Facility manager page  **Step5:** the system display view report page  **Step7: system display report** |
| Post condition | The Facility manager view report | |
| Alternative action | **A.** The system If no information found to see generated report  **1.** The system displays error message.  **2.** Go to **step4** to view report again. | |

Table 7 Approve/disapprove user

|  |  |  |
| --- | --- | --- |
| Use case id | **Use Case ID 7** | |
| Use case name | Approve/disapprove user | |
| Primary actors | Admin | |
| Description | **For Approving those who want to send maintenance request need.** | |
| Precondition | Admin Login, Maintenance checks the requirements. | |
| Basic flow of action | Actor action | System response |
| **Step1:** **Admin enter university id and password**  **Step4:** **Admin select approve request sender**  **Step6: Admin search request and**  **if the request is valid approve** | **Step2:** the system checks the authentication of university id and password  **Step3:** system display Admin  **Step5:** the system display approves  **Step7:** system check information  **Step8:** system display the request is approved |
| Post condition | Request sender login and send maintenance request needs | |
| Alternative action | **The request login request is disapproved** | |

Table 8 Assign maintenance technician

|  |  |  |
| --- | --- | --- |
| Use case id | **Use Case ID 8** | |
| **Use case name** | Assign maintenance technician | |
| Primary actors | **Facility manager** | |
| Description | Facility manager can assign the maintenance technician based on request that are submitted | |
| Precondition | Login, maintenance technician should be authenticated properly. | |
| Basic flow of action | **Actor action** | **System response** |
| **Step1:**  **Facility manager enter user name and password**    **Step4:** **Facility manager select view request.**  **Step6:** **Facility manager view request.** | **Step2:** the system checks the authentication of university id and password  **Step3:** system display Facility manager page  **Step5:** system display request records |
| Post condition | Facility manager views the submitted requests. | |
| Alternative action | **1.** The system displays error message.  **2.** Go to **step4** to view comment again. | |

Table 9 Track maintenance request

|  |  |  |
| --- | --- | --- |
| Use case id | **Use Case ID 9** | |
| **Use case name** | Track maintenance request | |
| Primary actors | **Facility manager** | |
| Description | Facility manager can Track the maintenance request which have requested but may be lost in any case. | |
| Precondition | Login, there must be a request from Request sender. | |
| Basic flow of action | **Actor action** | **System response** |
| **Step1:**  **Facility manager enter user name and password**    **Step4:** The Admin Track maintenance request.  **Step5:** The Admin Track maintenance request. | **Step2:** the system checks the authentication of university id and password  Step3**:** system display Facility manager page  **Step6:** The system gives conformation. |
| Post condition | Facility manager views the submitted requests. | |
| Alternative action | **1.** The system displays error message.  **2.** Go to **step4** to view comment again. | |

Table 10 Reject request

|  |  |  |
| --- | --- | --- |
| Use case id | **Use Case ID 10** | |
| **Use case name** | Reject request | |
| Primary actors | **Facility manager** | |
| Description | Facility manager can reject the maintenance request which is inappropriate or invalid based on request that are submitted | |
| Precondition | Login, there must be a request from Request sender. | |
| Basic flow of action | **Actor action** | **System response** |
| **Step1:**  **Facility manager enter user name and password**    **Step4:** **Facility manager select view request.**  **Step6:** **Facility manager view request.** | **Step2:** the system checks the authentication of university id and password  **Step3:** system display Facility manager page  **Step5:** system display request records  **Step7: Facility manager reject request.** |
| Post condition | Facility manager views the submitted requests. | |
| Alternative action | **1.** The system displays error message.  **2.** Go to **step4** to view comment again. | |

### State chart diagram

A state diagram consists of states, transitions, events, and activities. We use state diagrams to illustrate the dynamic view of a system. They are especially important in modeling the behavior of an interface, class, or collaboration. State diagrams emphasize the event-ordered behavior of an object, which is especially useful in modeling reactive systems.

We use state machines to model the behavior of any modeling element, although, most commonly, that will be a class, a use case, or an entire system which focuses on the event-ordered behavior of an object, which is especially useful in modeling reactive systems [6].

* Woldia university maintenance request tracking system has the following state diagrams: -

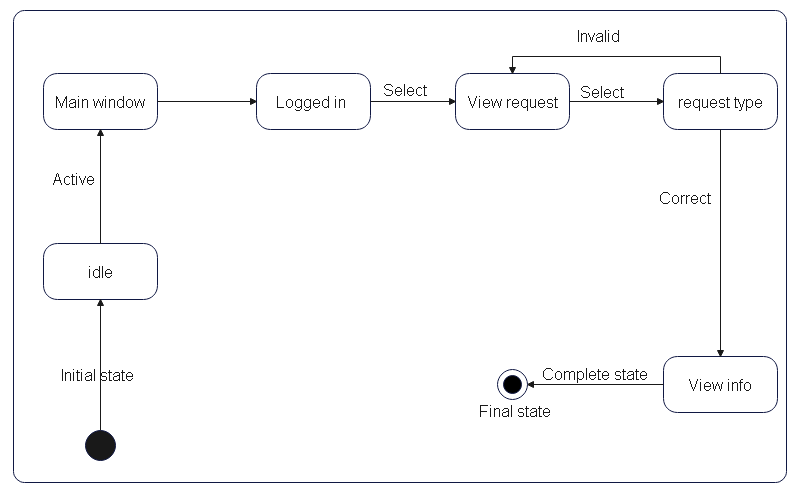


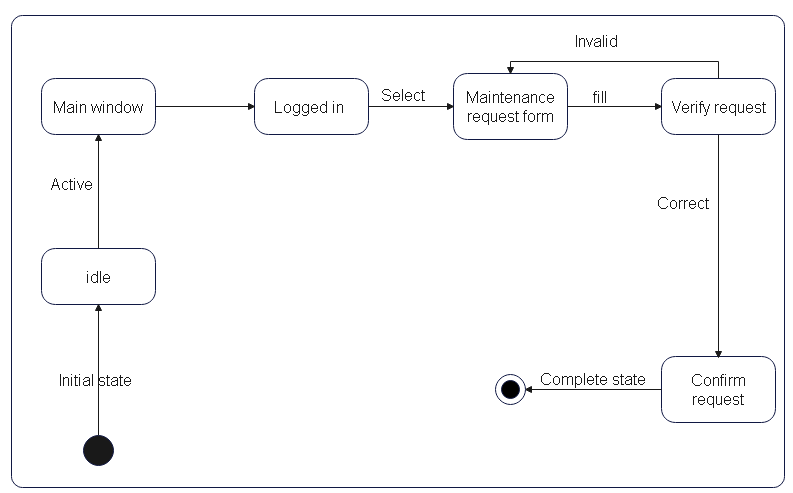
Figure 2 state chart for view request

Figure 3 State chart for maintenance request

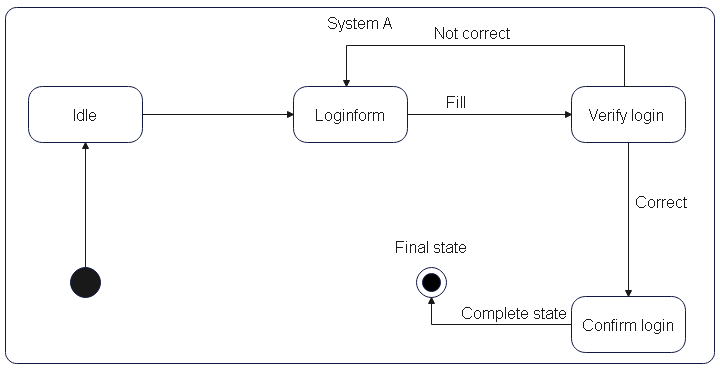


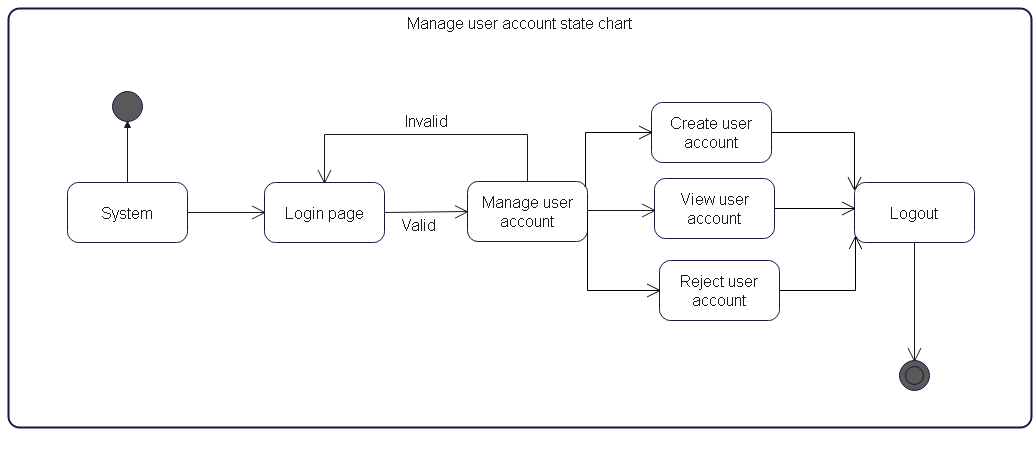
Figure 4 State chart for login

Figure 5 State chart for manage user accounts

### Activity Diagram

Activity diagrams describe the behavior of a class in response to internal processing rather than external events as in state chart diagram. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system [7].

* Woldia university maintenance request tracking system has the following Activity diagrams: -

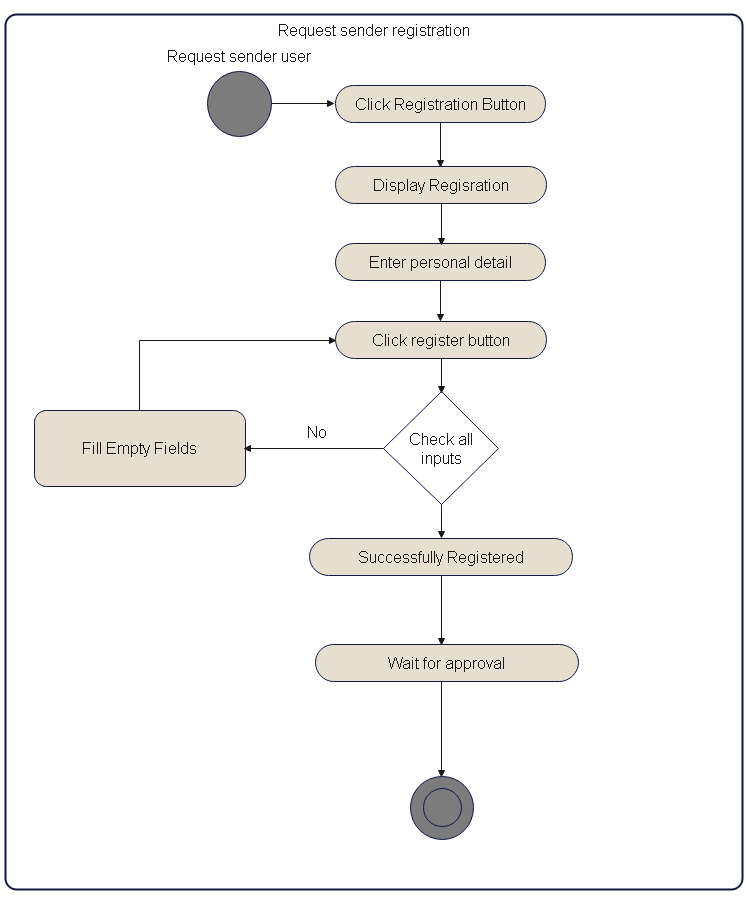


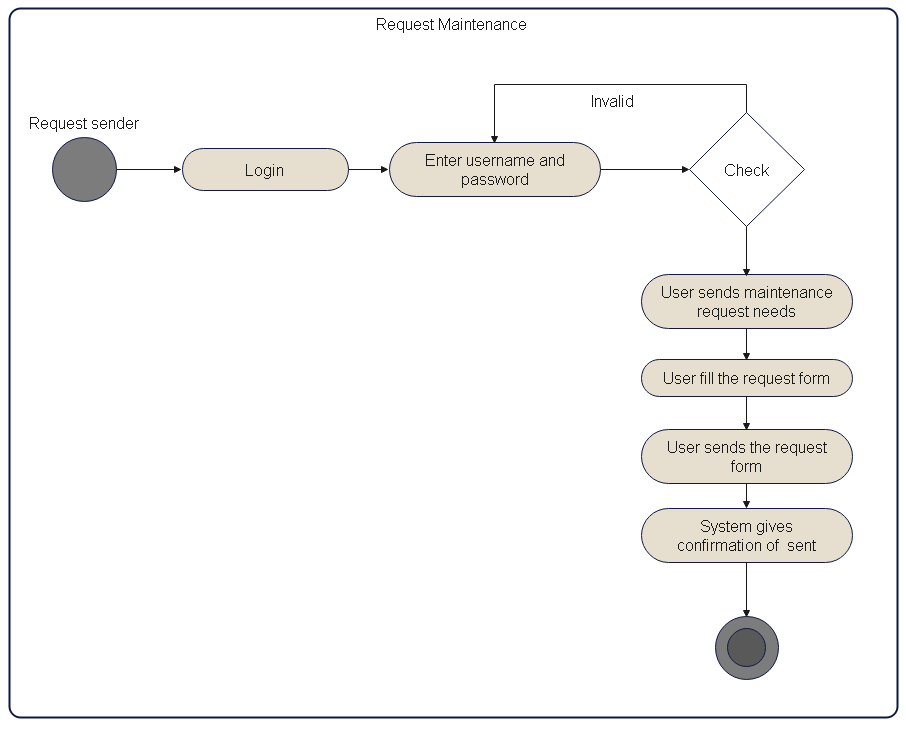
Figure 6 Activity diagram for request sender registration

Figure 7 Activity diagram for request maintenance

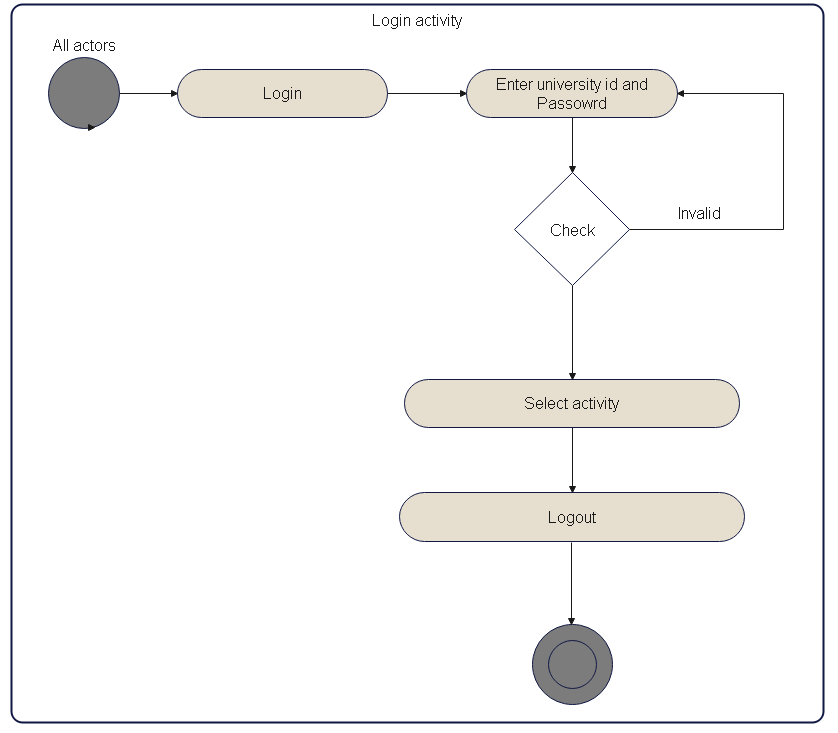


Figure 8 Activity diagram for login

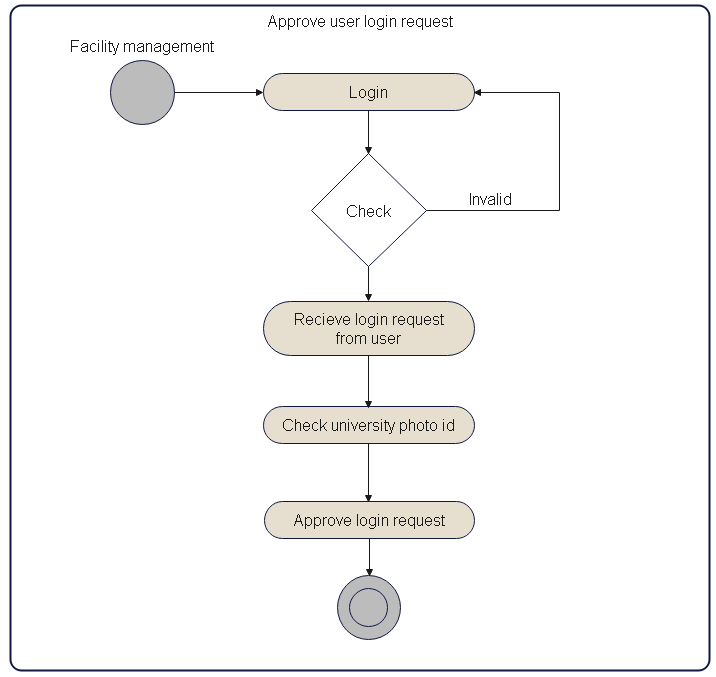


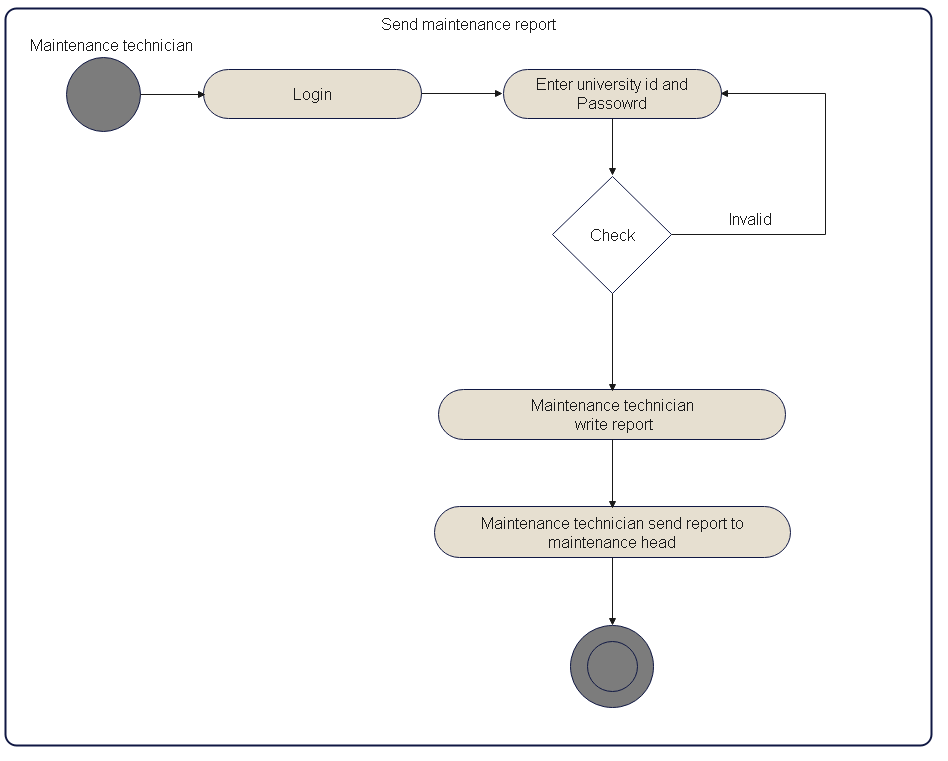
Figure 9 Activity diagram for Approve Login request

Figure 10 Activity diagram for send maintenance report

### Sequence diagram

UML is a kind of interaction diagram that shows how processes operate with one another and in what order. A sequence diagram shows object interactions arranged in time sequence. It shows the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario [8].

* Woldia university maintenance request tracking system has the following sequence diagrams: -

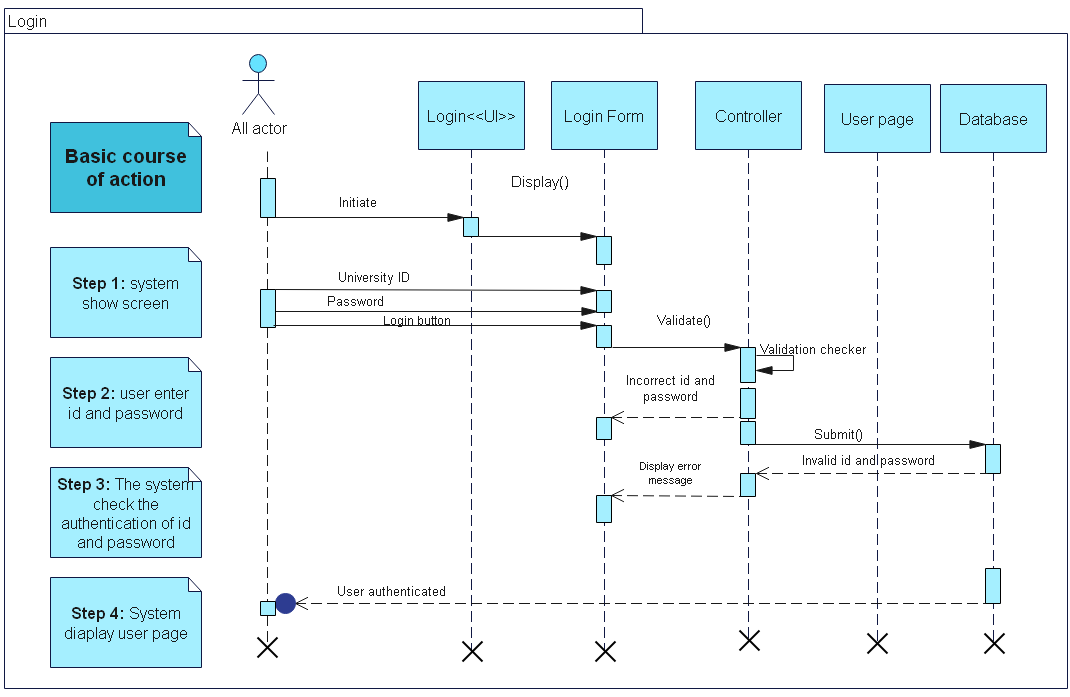


Figure 11 Sequence diagram for login



Figure 12 Sequence diagram for view request

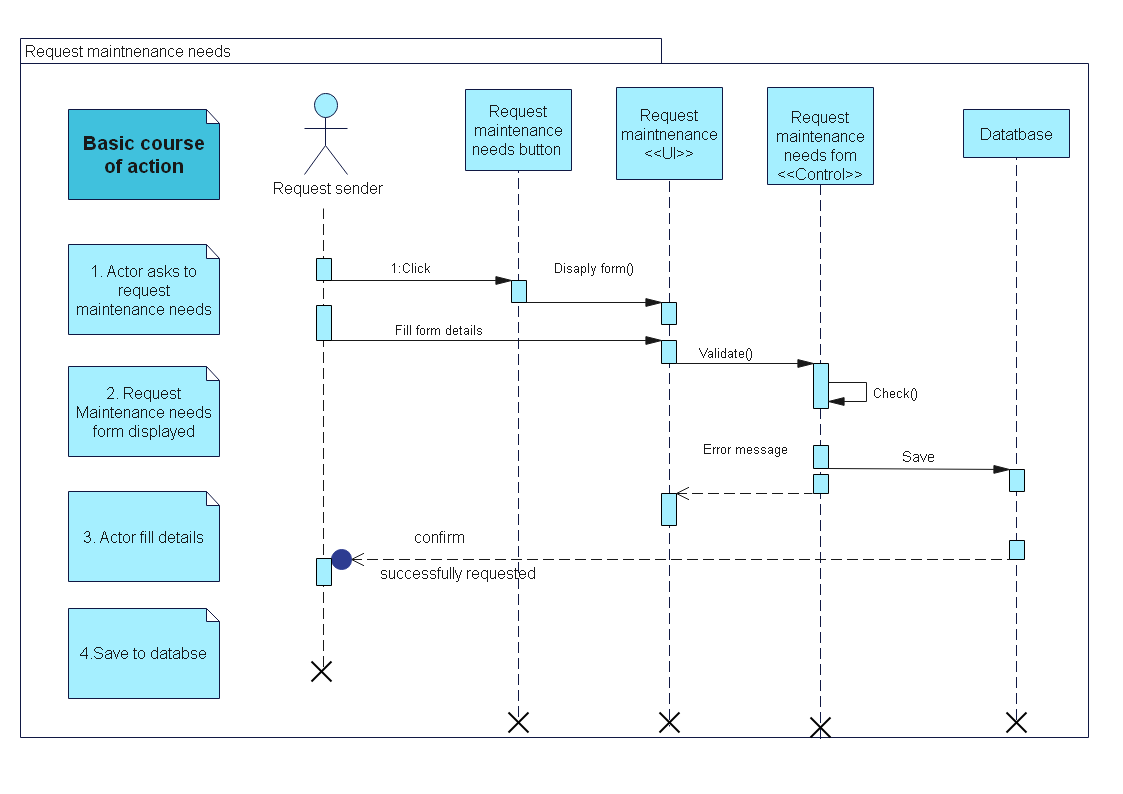


Figure 13 Sequence diagram for request maintenance needs

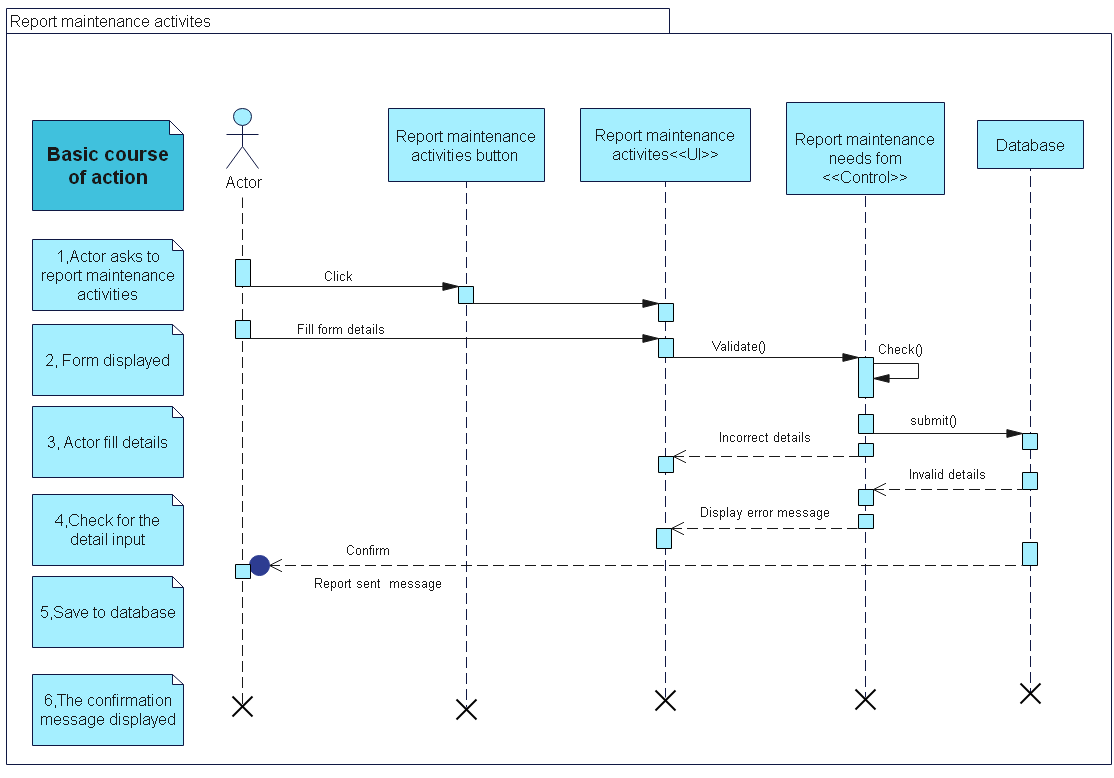


Figure 14 Sequence diagram for report maintenance activities

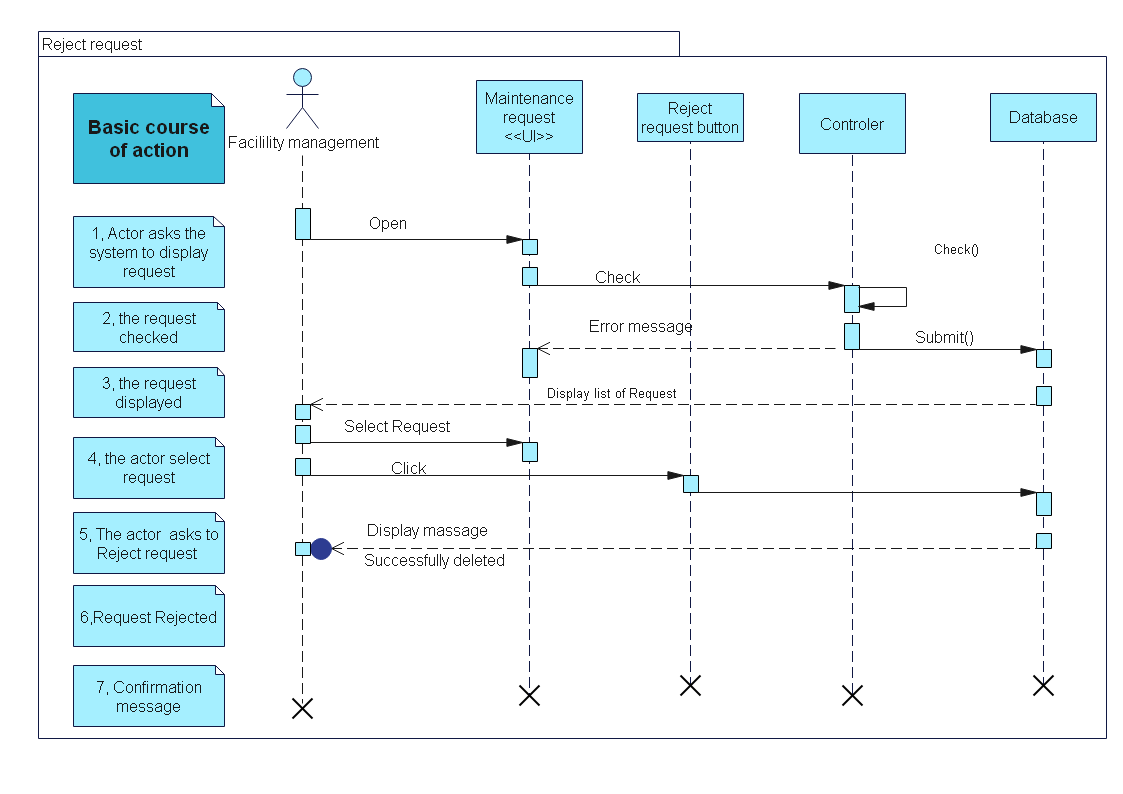


Figure 15 Sequence diagram for reject request

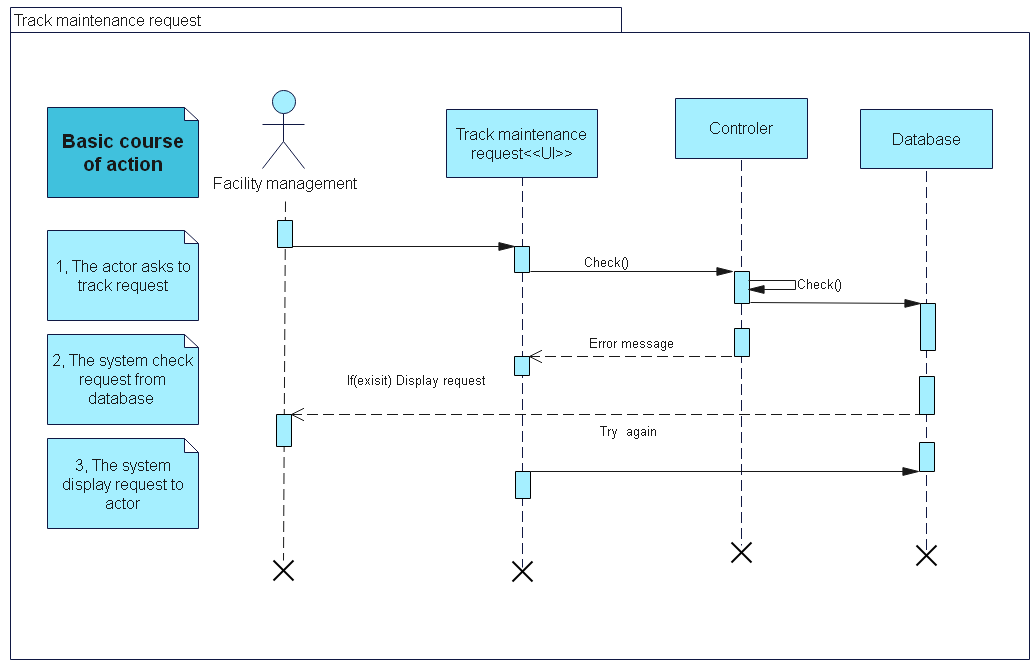


Figure 16 Sequence diagram for Track maintenance request

### Collaboration diagram

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behavior of a system.

The UML Collaboration diagram is used to model how objects involved in a scenario interact, with each object instantiating a particular class in the system. Objects are connected by links, each link representing an instance of an association between the respective classes involved. The link shows messages sent between the objects, and the type of message passed [9].



Figure 17 Collaboration diagram of login



Figure 18 Collaboration diagram of create account



Figure 19 Collaboration diagram of view request

### Conceptual modeling: Class diagram

Class diagram is static model that shows the classes and the relationships among classes that remain constant over phase the time. Class is the main building block of class diagram, which stores and manages information in the system. In the of conceptual class modeling, we just create or classes add their interrelationship [10].

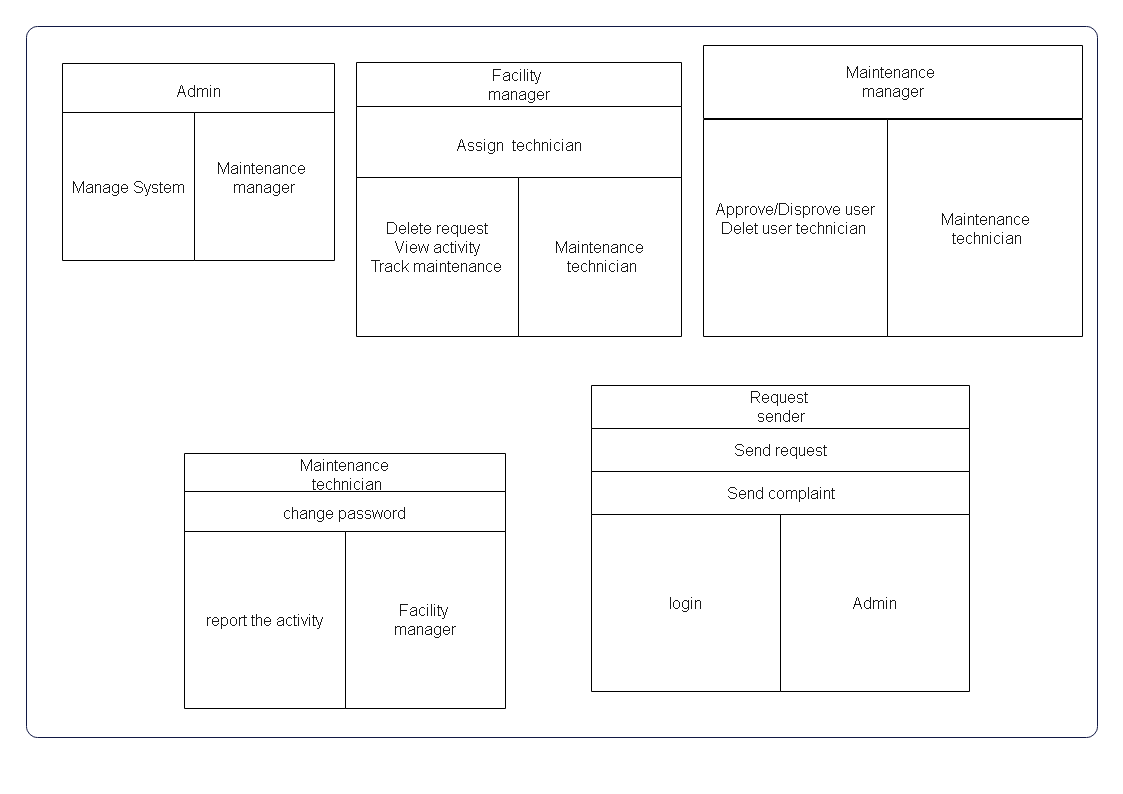


Figure 20 Class diagram

* 1. Key abstraction with CRC analysis

A CRC model is a collection of standard index cards that have been divided into three sections

* Class –collection of similar objects
* Responsibility –something that a class knows or does (contains attributes and methods)
* Collaborator –another class that a class interact with to fulfill its responsibilities



# **Chapter 3: System Design**

## Design goals

The design goal for our projects is to provide a user-friendly and efficient way for users to request maintenance need. The app should be designed to streamline the maintenance request process, making it easy for users to submit requests, assign requests, and receive updates on the progress of their requests.

Some specific design goals for our project includes:

* User-friendly interface: The app should be designed with a simple and intuitive interface that makes it easy for users to navigate and submit maintenance requests.
* Seamless integration with backend systems: The app should be designed to seamlessly integrate with backend systems.
* Real-time updates: The app should be designed to provide real-time updates to users on the status of their maintenance requests, including any changes in the requites status
* Automated notifications: The app should be designed to send automated notifications to users when their maintenance requests have been received, assigned, or completed.
* Self-service options: The app should be designed to allow facility manager to perform self-service tasks, such as canceling or assigning maintenance requests, without requiring assistance from customer service representatives.
* Very efficient work with low resource (memory) and time.
* To make the system easy to use
* Easy to manage
* To save loss of resource and time
* To reduce manual work with pen and paper

### Performance criteria

Performance requirements define acceptable response times for system functionality.

* Responsiveness: The app should respond quickly to user actions and inputs. The app should not take too long to load, and should not have any noticeable lag when performing tasks.
* Stability: The app should be stable and not crash or freeze frequently. It should handle errors and exceptions gracefully, and provide helpful error messages to users.
* Battery life: The app should not consume excessive battery power, as this can lead to user frustration and negative reviews. we should optimize the app's power usage to ensure it runs efficiently.
* Memory usage: The app should not consume excessive memory on the user's device, as this can slow down the device and cause other apps to crash. the app should optimized memory usage to ensure it runs smoothly.
* Network usage: The app should not consume excessive network data, as this can lead to user frustration and negative reviews.
* User experience: The app should provide a great user experience, with an intuitive and user-friendly interface. It should be easy to navigate, and provide helpful feedback to users.
* The system ever change retrieve as per user click through help of
* Good validations for user inputs will be done.
* Encrypted Password.
* Administrator has more rights than the sub users.

### Maintenance criteria

Maintenance is a critical aspect of our project is that to ensures the app remains functional and up-to-date over time. Here are some of the key maintenance criteria for our projects:

* Bug fixing: As users encounter bugs or issues within the app, it's important to fix them quickly to maintain user satisfaction and prevent negative reviews.
* Performance optimization: Over time, the app may become slower or less responsive due to changes in the operating system or other factors. we should monitor the app's performance and optimize it as needed to ensure it runs smoothly.
* Compatibility updates: As new versions of the Android operating system are released; the app may need to be updated to ensure it remains compatible. we should monitor for compatibility issues and release updates as needed to ensure the app continues to function properly.
* Security updates: As new security threats emerge, it's important to update the app to address these threats and protect user data. The development process should follow best practices for app security and release updates as needed to address any vulnerabilities.
* Feature updates: Over time, users may request new features or improvements to existing features. The development process should prioritize these requests and release updates as needed to keep users engaged and satisfied.
* During maintenance stage, the HRS can be referred for the validation.
* One who develop this should write correspondence documentation for maintain.

### End User Criteria

End users are the people who will be uses our project and their needs and expectations should be a primary consideration throughout the development process. User of the system should be able to know the criteria or responsibility of his /her chance to use this system.

Here are some of the key end user criteria that should be taken into account during our project development:

* User experience: The app should provide a great user experience, with an intuitive and user-friendly interface. It should be easy to navigate, and provide helpful feedback to users.
* Functionality: The app should provide the functionality that users need and expect. we should conduct user research to understand the needs and preferences of the target audience, and ensure that the app meets those needs.
* Compatibility: The app should work on a wide range of android devices and operating system versions, to ensure that all users can access and use the app. we should test the app on a variety of devices and Android versions to ensure compatibility.
* Create valid user when you use administrator
* Register as valid user.

### Reliability requirement

Reliability is an important requirement for our projects, as users expect apps to function consistently and without errors.

Some requirements that should be taken into account during our project development:

* Stability: The app should be stable and not crash or freeze frequently. It should handle errors and exceptions gracefully, and provide helpful error messages to users.
* Consistency: The app should provide consistent functionality across devices and operating system versions. Users expect apps to function the same way regardless of the device they are using.
* Testing: The app should be thoroughly tested to ensure that it meets reliability requirements. we should conduct both manual and automated testing to identify and fix any bugs or issues.
* Error handling: The app should handle errors and exceptions gracefully, and provide helpful error messages to users. Users expect to be informed of any errors or issues that occur within the app.
* Good validations for user inputs will be done.
* Avoid incorrect storage of records.

### Security requirement

Security is a critical requirement for our projects, as users expect their personal information and data to be kept safe and secure.

There are following key security requirements for our project development:

* Secure data storage: The app should store user data securely, using encryption and other security measures to prevent unauthorized access.
* Secure network communication: The app should use secure network protocols, such as HTTP, to prevent interception and tampering of data in transit.
* User authentication: The app should require strong user authentication, using methods such as passwords, or two-factor authentication, to prevent unauthorized access to user accounts.
* Authorization and access control: The app should implement strict authorization and access control policies, to ensure that users only have access to the data and features that they are authorized to use.
* Secure coding practices: The app should be developed using secure coding practices, such as input validation, to prevent common vulnerabilities.
* Security testing: The app should be thoroughly tested for security vulnerabilities, using tools such as penetration testing and vulnerability scanning, to identify and fix any potential issues.
* Encrypted Password.
* Administrator has more rights than the sub user.

## System architecture

System architecture is the means of designing the system based on class or layers. Layering is the concept of organizing the system software design into layers/collections of classes or components that fulfill a common purpose, such as implementing the user interface or the business logic of the system.



## Figure 4. 1 System architecture

## Table 4. 1 System architecture

|  |  |
| --- | --- |
| Layers | Descriptions |
| Interface (User Interface) layer | The interface layer is the layer in which the user relates with the system. |
| Process (application and controller) Layer | This layer consists of system interface and user interface (UI) classes that provide people access to our system. |
| Business (Domain) Layer | This layer implements and link the user data within the business objects, which means it validate the required data with the existing data for its correctness. |
| Data (Persistence) Layer | Persistence layers summarize the capability to store and delete data and it is the class layer in which all data about the user and the file are stored and accessed. |
| System Layer | This layer includes communicate all of the remaining four layers. |

## Subsystem Decomposition

To reduce the complexity of the system, we have decomposed the system into six parts based up on the functionality, where each those subsystems have their own subsystems.

Sub system for administrator

* Log in
* Create account
* Update account
* Make backup
* Log out
* Activate and deactivate user’s account

Sub system for Request sender

* Send Requests
* Send reports
* View requests

Sub system for Technician

* View assigned requests
* Send maintenance reports

Sub system for Facility manager

* View requests
* Track requests
* Assign maintenance technician to maintenances
* View reports
* Reject requests



## Component modeling

Systems may be built from components in component-based architecture. Component diagram shows how objects (classes) in your system are grouped together and form components. The components interact with each other either in giving service to other components or requesting service from another component. Show the relationship between software components, their dependencies and communications.



## Deployment Modeling

Deployment diagram show how the system is deployed on computers. In other words, it shows which component of the software is installed on which machine and how they communicate with each other if they are on different machines. Indicate where each component will be located, on what servers, machines or hardware. The deployment model should clearly show the physical communication links between hardware and software items like what runs where, etc.



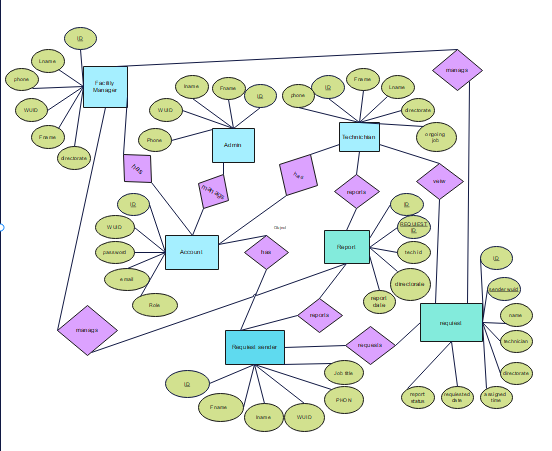
## Database Design

Database design is the process of producing a detailed data model of a database. This logical data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a Data Definition Language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity.

### Design ER diagram

An ER diagram, or entity-relationship diagram, is a graphical representation of entities and their relationships to each other in a database. It is used to model the data requirements of an organization or system, and to design a database schema that can be implemented in a relational database management system (RDBMS).





## User Interface Prototyping

User interface-flow diagram is called a navigation diagram boxes which represent major user interface elements, modeled as instances/objects, and the line represent the possible flow between them.



## Access control and security

### Access Control

Access control is way of limiting access to a system or to physical or virtual resources. In computing, access control is a process by which users are granted access and certain privileges to systems, resources or information.

Different actors have access to different information and data in the system. Access control and security specifies what the user can access or what cannot (limit to) perform by some users. This access control is verified (checked) by username and password. System represents an authenticated user. He/she uses login subsystem to check username and password that associated system user, but system user represents an authenticated user. They are used by login to subsystem to perform their activity on their belong page. The following table shows access control of our system.

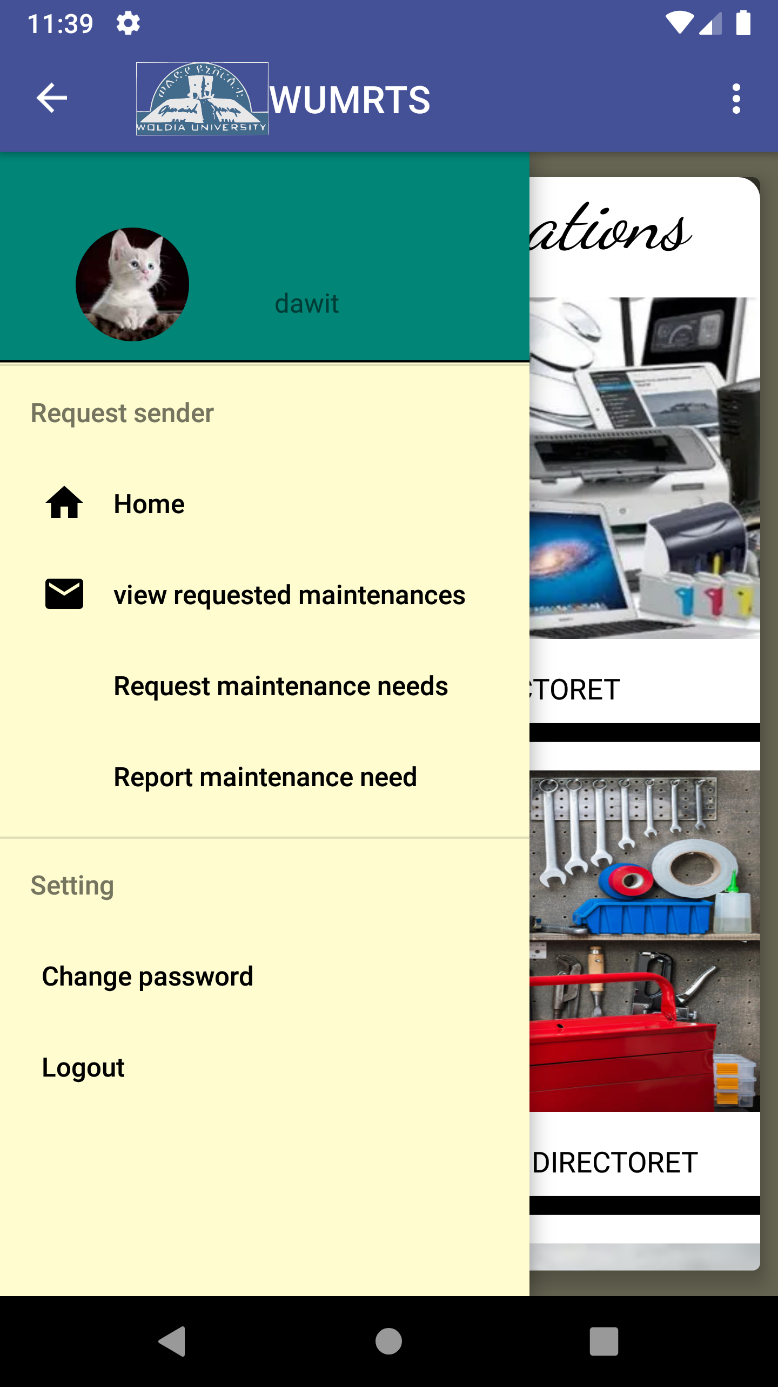
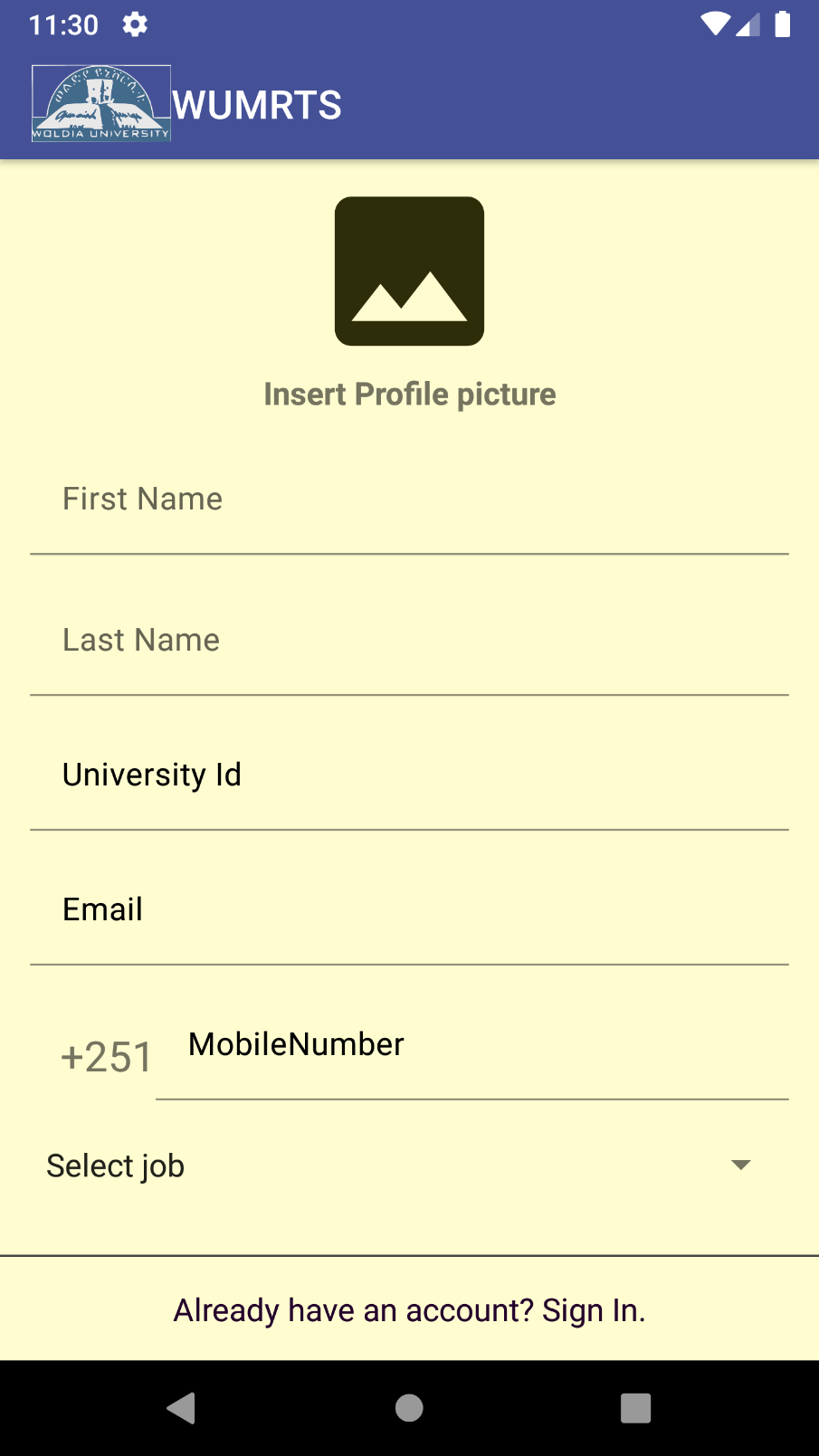
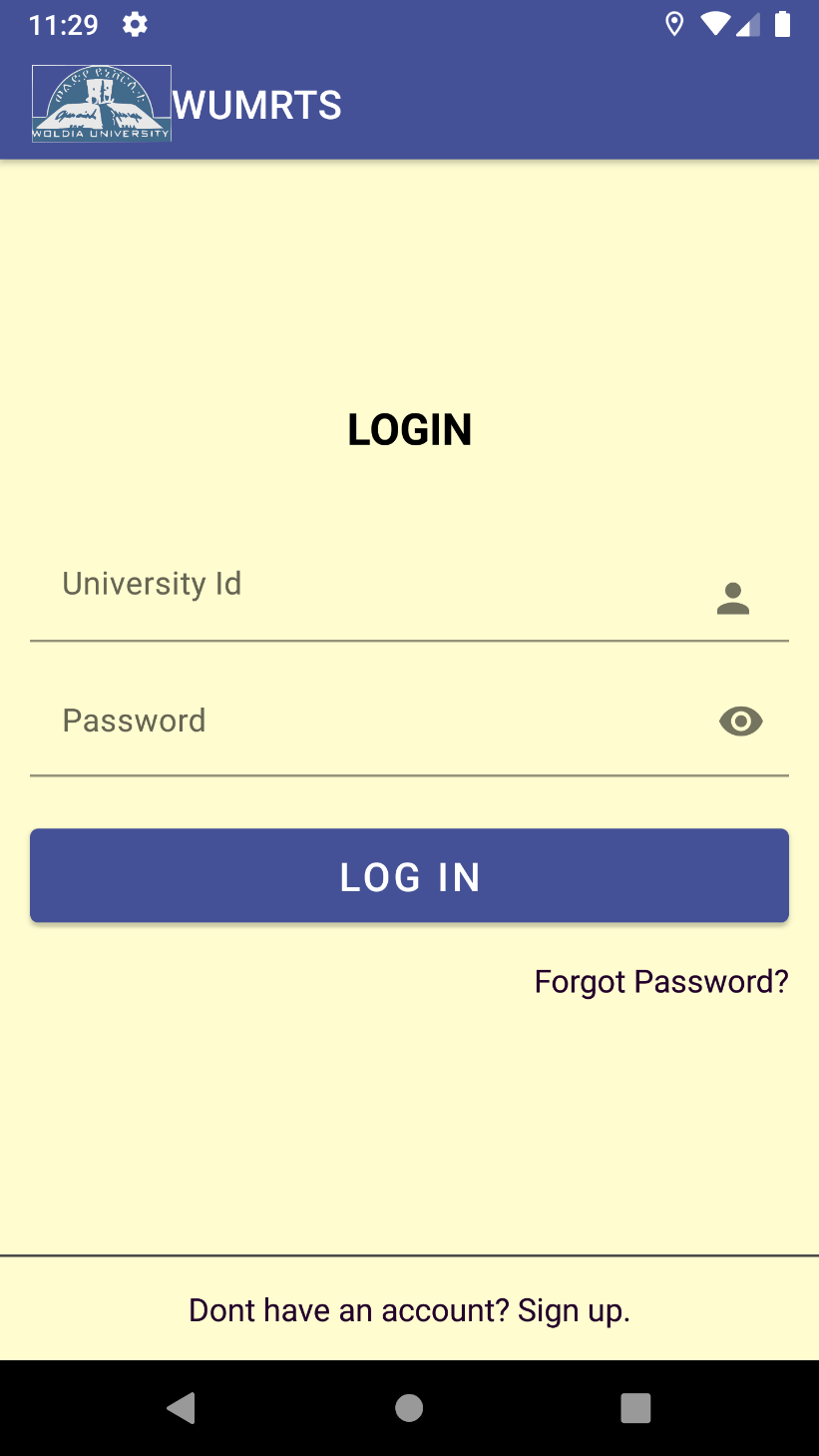
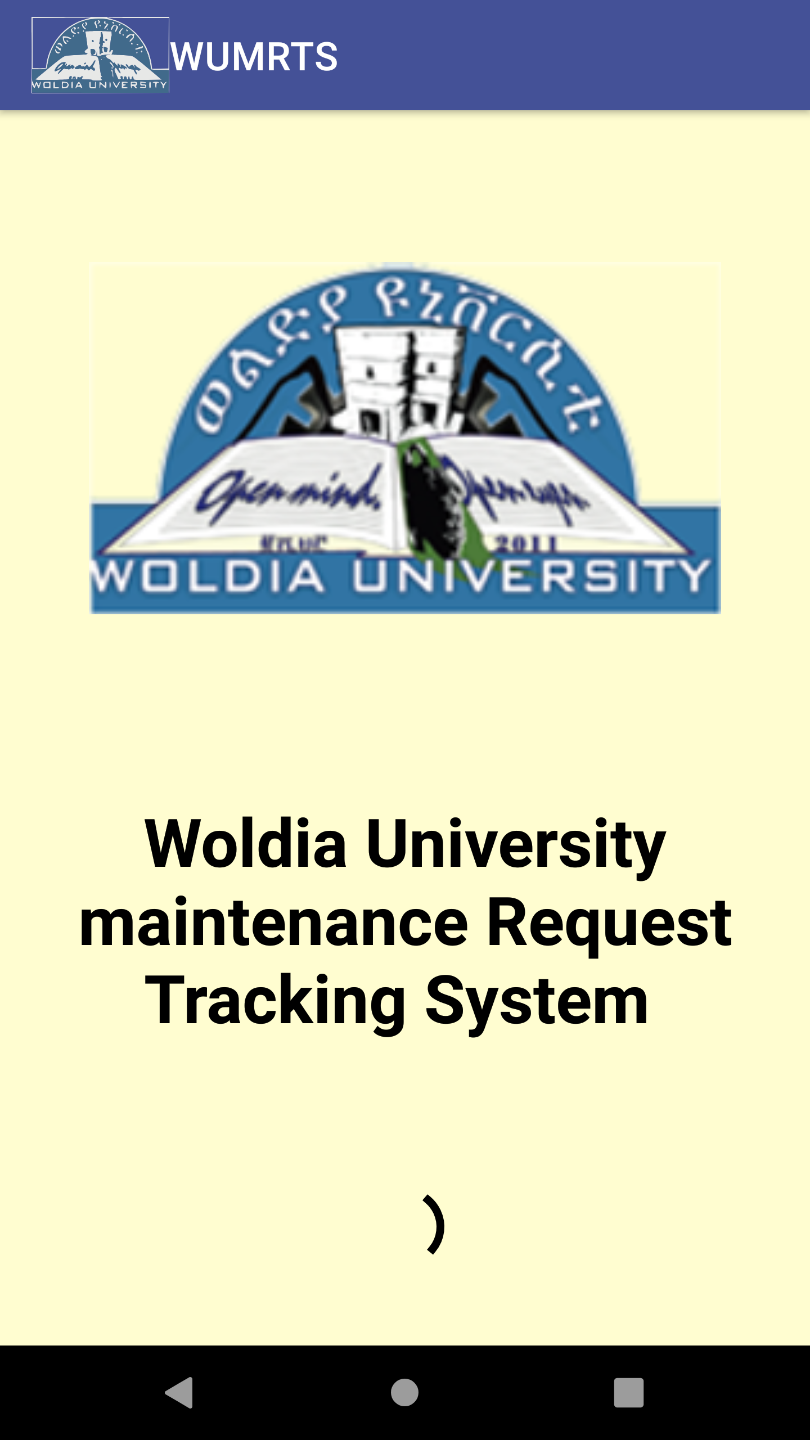
**Security Policy**

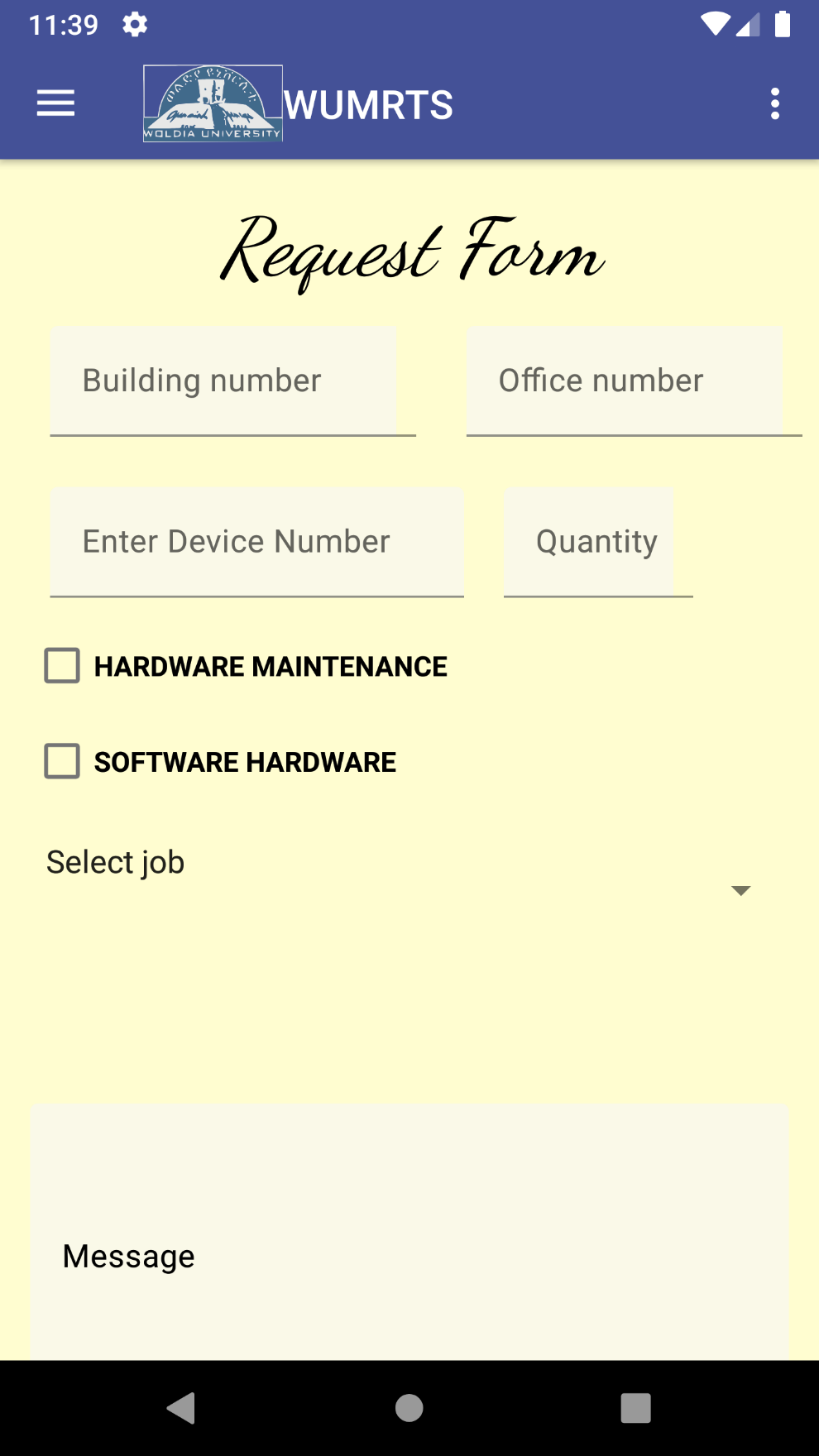
Since the system hold important project data and organizational data, the system is requiring strong security features to protect that valuable information. The system is highly secured by using User-Name and Password. To access the system, the user should have account.

|  |
| --- |
| **Functions** |
| **Admin** | **Facility management** | **Maintenance technician** | **Request sender** |
| Login | Yes | Yes | Yes | yes |
| Create account | Yes | No | No | No |
| Delete account | No | No | No | no |
| Active deactivate account | Yes | No | No | no |
| Change role rollaccount | Yes | No | No | no |
| Submit maintenance request | Yes | No | No | Yes |
| View complaint | No | Yes | No | no |
| Send report | Yes | No | Yes | no |
| Approve User login | Yes | Yes | No | no |
| Assign maintenance technician | No | Yes | No | no |
| Change password | Yes | Yes | Yes | Yes |
| Delete request | No | Yes | No | No |

# **Chapter Four: Implementation**

The implementation document helps users on how to work with the system. These phases transform the design and analysis phases of the system into a tangible system by writing the code. The following is the sample code for the implementation





<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="#FFFDD0"

>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/adminc"

android:text="LOGIN"

android:textStyle="bold"

android:textSize="22sp"

android:textColor="@color/black"

android:layout\_alignParentStart="true"

android:layout\_alignParentEnd="true"

android:gravity="center\_horizontal"

android:layout\_marginTop="120dp"

android:layout\_marginBottom="20dp"

/>

<com.google.android.material.textfield.TextInputLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/login\_email"

android:layout\_marginTop="15dp"

android:paddingEnd="15dp"

android:paddingStart="15dp"

android:layout\_below="@id/adminc">

<com.google.android.material.textfield.TextInputEditText

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:background="#00000000"

android:backgroundTint="#996A5E5E"

android:drawableRight="@drawable/ic\_baseline\_person\_24"

android:drawablePadding="10dp"

android:id="@+id/wuid"

android:hint="University Id"

android:inputType="textEmailAddress"

android:textColor="@color/black"

android:textColorHint="#A68A91" />

</com.google.android.material.textfield.TextInputLayout>

<!--\*\*\*\*\*\*\*password\*\*\*\*\*\*\*\*\*\*-->

<com.google.android.material.textfield.TextInputLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/login\_password"

android:layout\_marginTop="12dp"

android:paddingEnd="15dp"

android:paddingStart="15dp"

app:passwordToggleEnabled="true"

android:layout\_below="@id/login\_email">

<com.google.android.material.textfield.TextInputEditText

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:background="#00000000"

android:backgroundTint="#996A5E5E"

android:drawablePadding="10dp"

android:hint="Password"

android:id="@+id/pass"

android:inputType="textPassword"

android:textColor="@color/black"

android:textColorHint="@color/instacenterclord" />

</com.google.android.material.textfield.TextInputLayout>

<Button

android:id="@+id/login1"

android:layout\_width="383dp"

android:layout\_height="59dp"

android:layout\_below="@+id/login\_password"

android:layout\_gravity="center"

android:layout\_marginStart="15dp"

android:layout\_marginTop="20dp"

android:layout\_marginEnd="15dp"

android:text="Log In"

android:textColor="@color/white"

android:textSize="20sp" />

<TextView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/forgotpassword"

android:text="Forgot Password?"

android:gravity="end"

android:layout\_marginTop="12dp"

android:layout\_marginEnd="15dp"

android:textSize="16sp"

android:textColor="@color/dark\_purple"

android:layout\_below="@id/login1"/>

<TableRow

android:id="@+id/sr"

android:layout\_width="match\_parent"

android:layout\_height="1dp"

android:layout\_marginTop="12dp"

android:background="@color/textcolor"

android:layout\_above="@+id/signup"/>

<TextView

android:id="@+id/signup"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_alignParentBottom="true"

android:layout\_marginBottom="0dp"

android:gravity="center\_horizontal"

android:padding="15dp"

android:text="Dont have an account? Sign up."

android:textColor="@color/dark\_purple"

android:textSize="16sp" />

</RelativeLayout>

package com.example.wmrts;

import androidx.appcompat.app.ActionBar;

import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;

import android.app.Notification;

import android.app.NotificationChannel;

import android.app.NotificationManager;

import android.content.Context;

import android.content.Intent;

import android.content.SharedPreferences;

import android.media.Ringtone;

import android.media.RingtoneManager;

import android.net.Uri;

import android.os.Build;

import android.os.Bundle;

import android.os.Vibrator;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.Toast;

import org.json.JSONObject;

import java.util.HashMap;

import java.util.Locale;

import java.util.Map;

public class MainActivity extends AppCompatActivity {

Button login;

TextView signup,forgetpassword;

EditText ed1 ,ed2,ed123 ;

static String role = null;

public String ed11;

public SharedPreferences preferences;

public SharedPreferences.Editor editor;

@SuppressLint("MissingInflatedId")

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_login);

ActionBar actionBar = getSupportActionBar();

actionBar.setDisplayShowHomeEnabled(true);

actionBar.setIcon(R.drawable.trace);

preferences =getSharedPreferences("loginPrefs", Context.MODE\_PRIVATE);

String name = preferences.getString("Name", "");

String name1 = preferences.getString("wuid", "");

if (name.equalsIgnoreCase("admin"))

{Intent myIntent = new Intent(MainActivity.this, choose\_Admin.class);

MainActivity.this.startActivity(myIntent);

}else if (name.equalsIgnoreCase("mchead"))

{

Intent myIntent = new Intent(MainActivity.this, facilitymanagerchoose.class);

MainActivity.this.startActivity(myIntent);

}

else if (name.equalsIgnoreCase("mcexpert"))

{

Intent myIntent = new Intent(MainActivity.this, choosetechician.class);

myIntent.putExtra("wuid", name1);

MainActivity.this.startActivity(myIntent);

}

else if (name.equalsIgnoreCase("client"))

{

Intent myIntent = new Intent(MainActivity.this, statuscheck.class);

ed1 = findViewById(R.id.wuid);

ed11= ed1.getText().toString();

myIntent.putExtra("wuid", name1);

MainActivity.this.startActivity(myIntent);

}

ed123 = findViewById(R.id.wuid);

login = findViewById(R.id.login1);

signup =findViewById(R.id.signup);

forgetpassword =findViewById(R.id.forgotpassword);

forgetpassword.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent myIntent = new Intent(MainActivity.this, forget\_password.class);

MainActivity.this.startActivity(myIntent);

}

});

signup.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent myIntent = new Intent(MainActivity.this, register.class);

MainActivity.this.startActivity(myIntent);

}

});

login.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

ed1 = findViewById(R.id.wuid);

ed2 = findViewById(R.id.pass);

String ed11 = ed1.getText().toString();

String ed22 = ed2.getText().toString();

if (ed11.equals("") ) {

ed1.setError("This field is required");

Toast.makeText(MainActivity.this, "Insert id", Toast.LENGTH\_SHORT).show();

} else if (ed22.equals("")) {

ed2.setError("This field is required");

Toast.makeText(MainActivity.this, "Insert password", Toast.LENGTH\_SHORT).show();

} else

{

insertData();

retrieveData();

}

}

});

}

private void insertData() {

StringRequest request = new StringRequest(Request.Method.POST, "http://192.168.43.228:80//wumrts/login.php",

new Response.Listener<String>() {

@Override

public void onResponse(String response) {

String g =response.toLowerCase(Locale.ROOT).trim();

String y =response.toLowerCase(Locale.ROOT);

preferences =getSharedPreferences("loginPrefs", Context.MODE\_PRIVATE);

editor = preferences.edit();

if(g.equals("maintenance\_manager")){

editor.putString("Name","admin");

editor.apply();

Intent myIntent = new Intent(MainActivity.this, choose\_Admin.class);

Toast.makeText(MainActivity.this, "Logged", Toast.LENGTH\_SHORT).show();

MainActivity.this.startActivity(myIntent);

} else if(g.equals("mchead")){

editor.putString("Name","mchead");

editor.apply();

Intent myIntent = new Intent(MainActivity.this, facilitymanagerchoose.class);

Toast.makeText(MainActivity.this, "Logged", Toast.LENGTH\_SHORT).show();

MainActivity.this.startActivity(myIntent);

}

else if(g.equals("mcexpert")){

ed1 = findViewById(R.id.wuid);

ed11= ed1.getText().toString();

editor.putString("Name","mcexpert");

editor.putString("wuidexpert",ed11);

editor.apply();

Intent myIntent = new Intent(MainActivity.this, choosetechician.class);

Toast.makeText(MainActivity.this, "Logged", Toast.LENGTH\_SHORT).show();

myIntent.putExtra("wuid", ed11);

MainActivity.this.startActivity(myIntent);

}

else if(g.equals("client")){

editor.putString("Name","client");

editor.apply();

Intent myIntent = new Intent(MainActivity.this, statuscheck.class);

Toast.makeText(MainActivity.this, "Logged", Toast.LENGTH\_SHORT).show();

String keyIdentifer = null;

ed1 = findViewById(R.id.wuid);

ed11= ed1.getText().toString();

myIntent.putExtra("wuid", ed11);

editor.putString("wuid",ed11);

editor.apply();

retrieveData();

MainActivity.this.startActivity(myIntent);

}

else{

Toast.makeText(MainActivity.this, response, Toast.LENGTH\_SHORT).show();

}

}

}, new Response.ErrorListener() {

@Override

public void onErrorResponse(VolleyError error) {

Toast.makeText(MainActivity.this, error.getMessage(), Toast.LENGTH\_SHORT).show();

}

}

){

@Override

protected Map<String, String> getParams() throws AuthFailureError {

Map<String,String> params = new HashMap<String,String>()

ed1 = findViewById(R.id.wuid);

ed2 = findViewById(R.id.pass);

String ed11= ed1.getText().toString();

String ed22= ed2.getText().toString();

params.put("wuid",ed11);

params.put("password",ed22);

preferences =getSharedPreferences("loginPrefs", Context.MODE\_PRIVATE);

editor = preferences.edit();

editor.putString("wuid",ed11);

editor.apply();

return params;

}

};

RequestQueue requestQueue = Volley.newRequestQueue(MainActivity.this);

requestQueue.add(request)

}

}

<?php

$connection = mysqli\_connect("localhost","root","","wumrts");

$un = $\_POST['wuid'];

$pw = $\_POST['password'];

$result = array();

$result['data'] = array();

$encoded= md5($pw);

$select= "SELECT \*from account where wuid = '".$un."' AND status='Allowed'";

$responce = mysqli\_query($connection,$select);

if (mysqli\_num\_rows ( $responce )>=1){

while($row = mysqli\_fetch\_array($responce))

{

$er=$row['2'];

if($er==$encoded){

$index['id'] = $row['0'];

$index['wuid'] = $row['1'];

$index['password'] = $row['2'];

$index['role'] = $row['3'] if($encoded==$index['password']) {

// array\_push($result['data'], $index);

if($index['role'] =='Maintenance Manager')

{

echo "maintenance\_manager";

} else if($index['role'] =='Technician'){

echo "mcexpert";

}

else if($index['role'] =='Facility Manager'){

echo "mchead";

}

else if($index['role'] =='client'){

echo "client";

}

}

$result["success"]="1";

mysqli\_close($connection);

}

else{

echo "Incorrect password";

} } }else

{

echo "Incorrect username";

}

?>

# **Chapter Five: Testing**

## Testing

Final phase of implementation is testing. Software testing isthe process of analyzing a software item to detect the differences between existing and required conditions (that is, bugs) and to evaluate the features of the software item. Software testing is an activity that should be done throughout the whole development process.

Testing is a process to show the correctness of the program. Testing is checking of the system workability in an attempt to discover errors and avoiding such errors from the system. In this the team members tested the entire system as a whole with all forms, code, modules. In this we tested all the functionalities in the System. All errors in the forms, functions, modules have been tested. The following are different testing strategies.

* **Unit testing:-**Every module of the System is separately tested. I.e. the team tests every module by applying some selection mechanism. Through this mechanism every modules gets tested. If an error occurs correction will be taken without affecting another module.
* **Integrated Testing: -**In this testing part, all the modules will be combined together and tested it for its fitness with each other and with the systems functionality. If error occurs in combining them, the module with problem will be identified and recombined.
* **Black Box Testing: -**The team has conducted this testing procedure to evaluate only the outputs generated in response to selected inputs and execution conditions.
* **White Box Testing: -**The team has conducted this testing procedure during writing the code for each desired components of the system to check if the written code is working properly or not.
* **Functional (Black Box Testing) and System Testing: -** In this testing, the team performs over all functional testing by checking whether it meets or not meets the required target.
* **Compatibility Testing:-**
  + Hardware Compatibility test- the system is compatible with all the Hardware and Software listed under the Hardware and Software Acquisitions.
  + Software compatibility test – the system is compatible with all the software listed under the development tools table.
* **User Interface Testing:**- the team has conducted this testing procedure to evaluate the GUI elements like field forms , Spinners, input type length, radio button are work properly and suitable for the users. As a result all of these components are working properly.
* **Usability Testing:** -The team has conducted this testing procedure to evaluate the extent to which a user can learn to operate, prepare inputs for, and interpret outputs of a system or component and the system’s user friendless.

## Features to be Tested or not to be Tested

The comprehensive analysis and test of the software during all stages of its development and maintenance in parallel with the development process can only be achieved if the system features to be tested or not to be tested are clearly identified. Hence, the following are list of the features to be tested and not to be tested.

### Features tested

* Functional and non-functional requirements of the software tested using the system testing strategy.
* Design goals, design to code correlation and installation testing made.
* Classes and their methods that are participants in use cases will be tested using system testing strategy. An integration testing made after each class is tested independently and it keeps testing combination of classes until the entire system has been integrated.
* Screen messages, confirmations, warnings and errors, will be tested if they are correct and clear.
* Required and optional fields tested if they are properly processed. Consistency of screen names also tested.
* Database connection with the system tested.

# **Chapter six: Conclusions and Recommendation**

## Conclusions

As far we were intended in analyzing the existing system of Woldia University Maintenance request tracking system to our automate system that solves the difficulties related to the existing system.

The mobile application system has been one of the fastest growing systems in the world and it

Is still growing fast. Mobile application system is one of the wireless technologies that have been rising in The world. Mobile application system is an application that lives on a mobile phone.

Woldia University Maintenance request tracking system is helps workers to request for maintenances sat anywhere in every time through the use of internet and other additional technologies if they are only member of campus. Through this project the Request senders can request for maintenances by entering required information’ .

## Recommendations

The team member has a recommendation towards the project for modification of some features to make the system perfect and more efficient. So anyone who has the intension to register,

In addition to these, any user to use this system, he/ she have to know some knowledge how to use this system. Among the following is the most limitation to obstacles to our project.

* Android studio is new version so it is difficult to install.
* The personal computer is less amount of version to hold the android application.
* lack of connection
* android application is new for us, so difficult to easily understand

By such and other unlisted reasons our project is not fully completed. We give best opportunity for the next generation. We hope the coming generation work efficiently and effectively to apply this system.

# References

|  |  |
| --- | --- |
| [1] | tzadmission, "Woldia University Historical Background," [Online]. Available: https://tzadmission.net/woldia-university-historical-background/. [Accessed 4 2 2023]. |
| [2] | E. S. P. f. E. &. Career, "Woldia University: Ethiopia," Woldia University: Ethiopia, [Online]. Available: https://www.neaea.com/woldia-university/?nowprocket=1. [Accessed 6 2 2023]. |
| [3] | javatpoint, "Iterative Model," [Online]. Available: https://www.javatpoint.com/software-engineering-iterative-model. [Accessed 12 1 2023]. |
| [4] | investopedia, "Feasibility Study," [Online]. Available: https://www.investopedia.com/terms/f/feasibility-study.asp. [Accessed 3 3 2023]. |
| [5] | IBM, "What are business rules," [Online]. Available: https://www.ibm.com/topics/business-rules. [Accessed 3 1 2023]. |
| [6] | v. paradigm, "All You Need to Know about State Diagrams," 6 3 2023. [Online]. Available: All You Need to Know about State Diagrams. |
| [7] | T. point, "UML - Activity Diagrams," Tutorials point, 6 3 2023. [Online]. Available: https://www.tutorialspoint.com/uml/uml\_activity\_diagram.htm. [Accessed 6 3 2023]. |
| [8] | U. M. L. (. |. S. Diagrams, "geeks for geeks," [Online]. Available: https://www.geeksforgeeks.org/unified-modeling-language-uml-sequence-diagrams/. [Accessed 5 2 2023]. |
| [9] | K. McNeish, "UML Collaboration Diagrams," [Online]. Available: https://www.codemag.com/article/0205051/UML-Collaboration-Diagrams. [Accessed 5 2 2023]. |
| [10] | G. f. geeks, "Unified Modeling Language (UML) | Class Diagrams," [Online]. Available: https://www.geeksforgeeks.org/unified-modeling-language-uml-class-diagrams/. [Accessed 5 3 2023]. |

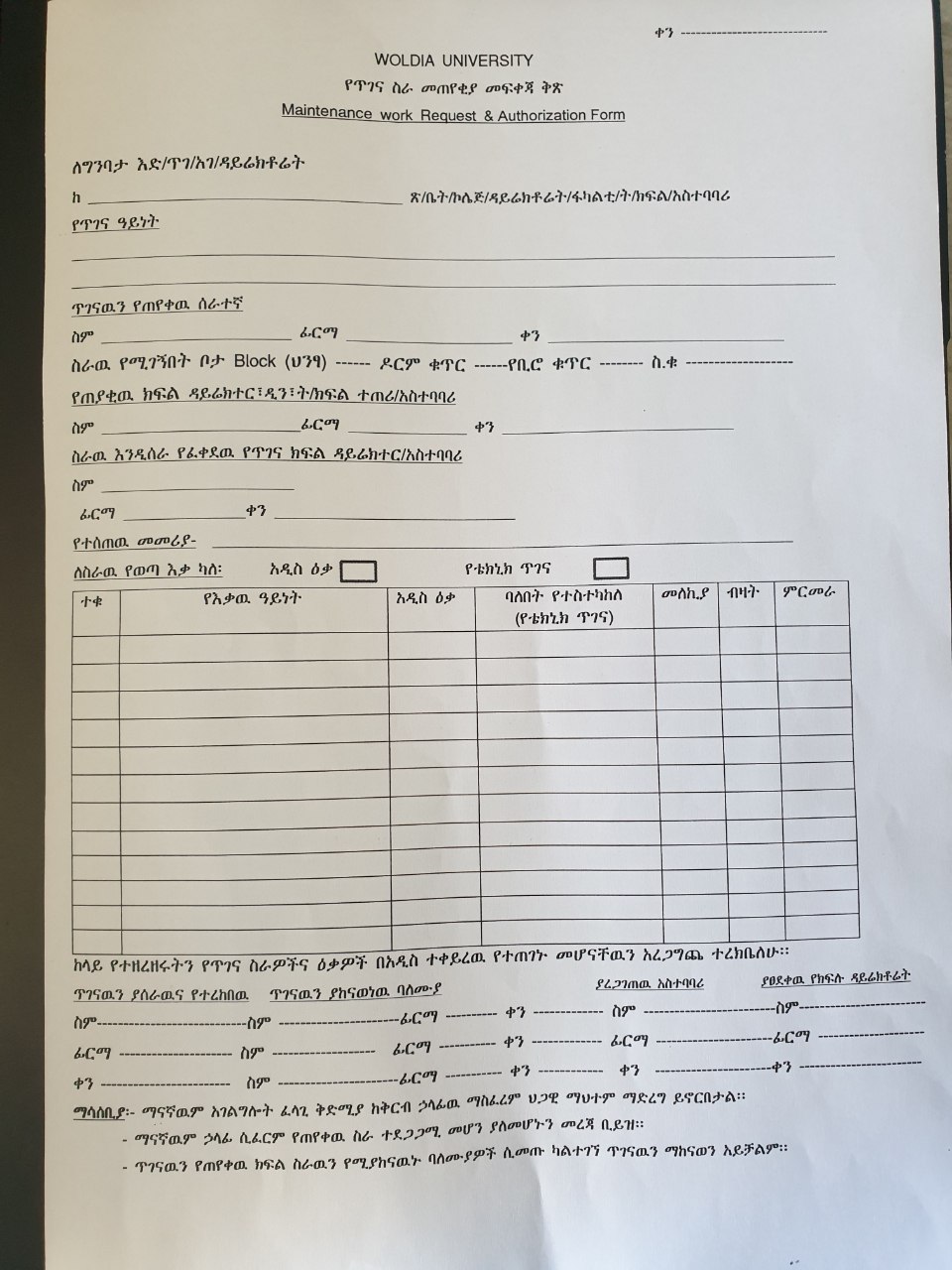
# **Appendix**

Question we asked during an interview:

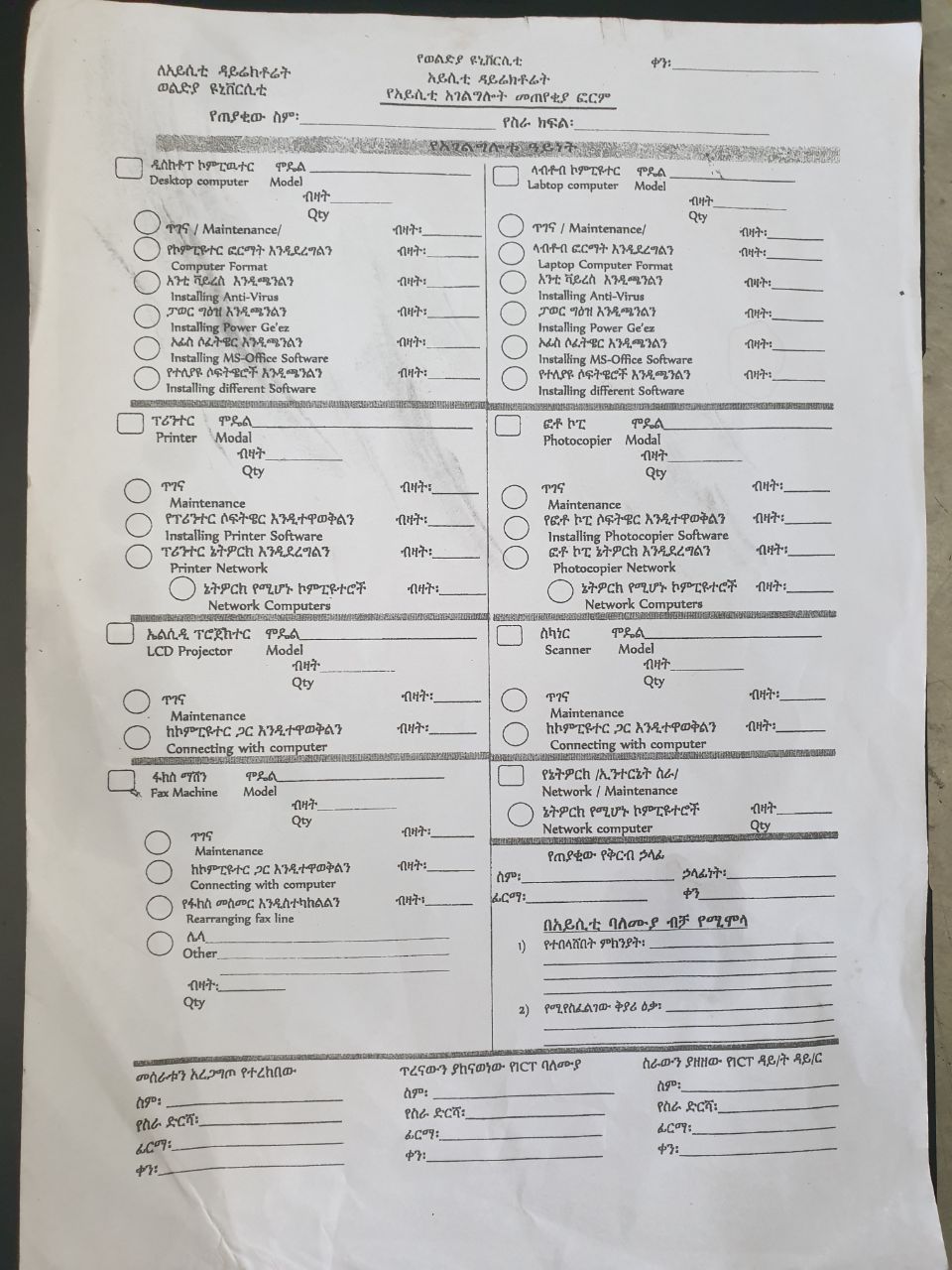
1. Do you have a formal documented maintenance practice and procedures that includes standard times that the maintenance staff is to meet?
2. Does these practices and procedures include a written procedure describing the process along with a time standard?
3. Do you have a quality assurance group to ensure maintenance activity and procedures are producing a quality product?
4. What is your schedule for a preventive maintenance program?
5. What is the classification of personnel and technician and maintenance technician, and what level of training is required?
6. What is the classification of your maintenance personnel at these locations? List all qualifications including the training required.
7. Have you modified your training programs to keep your personnel up to speed on the new equipment and technology on the recently purchased in university?

Documents we have gathered for document analysis are listed below:

1. This document is used collect maintenance request for building , offices , library etc. Also, it is used report activities done and material used.



1. This document is used to collect maintenance requests related to ICT needs (laptops printers desktops scanners etc.)



1. This document is used to collect maintenance request needs for dormitory.

