  
UNIVERSITY OF GONDAR  
FACULTY OF INFORMATICS  
DEPARTMENT OF COMPUTERSCIENCE  
**WEB BASED FINAL YEAR INDUSTIRAL PROJECT MANAGEMENT SYSTEM**

INDUSTRIAL PROJECT PREPARED BY:

NAME OF STUDENTS IDNO

1. ATSEDE WORKU……………………..GUE/6793/07
2. KALKIDAN ASMARE………………...GUE/6868/07
3. TADESSE ALENE…...……………..…GUE/6928/07
4. GEBIYANESH SISAY…………………GUE/6846/07
5. WELETESENBET TSEGA…………….GUE/6943/07

UNDER THE GUIDANCE OF

INSTRUCTOR ANTENEH MEKURIAW

SUBMMATION DATE

07/03/2019G.C

**DECLARATION**

This is to declare that this project work which is done under the supervision of INSTRUCTOR ANTEHEN MEKURIYAW and having the title FINAL YEAR INDUSITRIAL PROJECT MANAGEMENT SYSTEM is the sole contribution of:

**GROUP MEMEBERS SIGNATURE**

ATSEDE WORKU ...…………….………

KALKIDAN ASMARE …………….…………

TADESSE ALENE ………………………..

GEBIYANESH SISAY ……………………….

WELETESENBET TSEGA …………………….…

No part of the project work has been reproduced illegally (copy and paste) which can be considered as Plagiarism. All referenced parts have been used to argue the idea and have been cited properly. We will be responsible and liable for any consequence if violation of this declaration is proven.

Date 07/03/2019G/C

**CERTIFICATE**

I certify that this BSc. industrial project titled WEB BASED FINAL YEAR INDUSIRIAL PROJECT MANAGEMENT SYSTEM by: Atsede Worku, Kalkidan Asmare, Weletsenbet Tsega, Gebiyansh sisay, Tadese Aleneis approved by me for submission. I certify further that, to the best of my knowledge, the report represents work carried out by the students.

Advisor Name INSTRUCTOR ANTENEH MEKURIYAW

07/03/2019 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date Signature

# **ACKNOWLEDGEMENT**

Any accomplishment requires the blessing of the God. So the first thanks goes to God. We have taken efforts in this project. However, it would not have been possible without the kind of support and help of many individuals. Then, we would like to thanks all those who contributed throughout the completion of our project and helped us with valuable suggestion for the improvement of the system. We are extremely grateful for our advisor instructor Antenehe Mekuryaw for guiding us throughout the completion of the project. The next gratitude goes to our academic institution, university of Gondar by providing service to complete this industrial project in very well. Moreover we would like to thanks our group members for their own participation for the industrial project.

# **ACRONOMY**

G.C: Gregorian calendar

E.C: Ethiopian Calendar

UML: Unified modeling language

SDLC: System Development Life Cycle

HTML: Hypertext markup language

HTTP:Hper text transfer protocol

PHP: Hypertext preprocessor

CSS: Cascading Style Sheet

MYSQL: My Standard query language

WAMP: Window, Apache, MYSQL, PHP

GB: Gigabyte

RAM: Random Access memory

CD: Compact Disk

GUI: Graphical User Interface

GPA: Grade point Average

CPU: Central Processor Unit

HZ: Hertz

GHZ: Gigahertz

MB: Megabyte

# **ABRIVAION**

FYP: Final Year Project

PHC: Public Health College

TC: Training Center

GCMS: Gondar College of medical science

UC: Use Case

STUDID: Student Id

EXAID: Examiner Id

ADVID: Advisor Id

DEPID: Department head Id

COOID: Coordinator Id

COMID: Comment Id

PROGID: Progress Id

Table of Contents

[**ACKNOWLEDGEMENT** iii](#_Toc3874078)

[**ACRONOMY** iv](#_Toc3874079)

[**ABRIVAION** v](#_Toc3874080)

[**LIST OF FIGURE** ix](#_Toc3874081)

[**LISTOF TABLE** x](#_Toc3874082)

[**ABSTRUCT** xii](#_Toc3874083)

[**1.** **Introduction** 1](#_Toc3874084)

[**1.1.** **Background of the study** 1](#_Toc3874085)

[**1.2. Statement of the problem and justification** 2](#_Toc3874086)

[**1.3. Objective of the study** 3](#_Toc3874087)

[**1.3.1. General objective** 3](#_Toc3874088)

[**1.3.2. Specific objective** 3](#_Toc3874089)

[**1.4. Scope of the project** 3](#_Toc3874090)

[**1.5. Limitation of the project** 3](#_Toc3874091)

[**1.6. System development methodology** 4](#_Toc3874092)

[**1.7. Feasibility study** 7](#_Toc3874093)

[**1.7.1. Technical feasibility** 7](#_Toc3874094)

[**1.7.2.** **Economic feasibility** 7](#_Toc3874095)

[**1.7.3.** **Operational feasibility** 7](#_Toc3874096)

[**1.8.** **Significance of the project** 8](#_Toc3874097)

[**1.9.** **Beneficiaries** 8](#_Toc3874098)

[**1.10.** **Project organization** 9](#_Toc3874099)

[**1.11.** **Time schedule** 9](#_Toc3874100)

[**CAPTER TWO** 10](#_Toc3874101)

[**2.** **Requirement analysis** 10](#_Toc3874102)

[**2.1. Introduction** 10](#_Toc3874103)

[**2.2. Current system description** 10](#_Toc3874104)

[**2.2.1. Major functions of the current system** 10](#_Toc3874105)

[**2.3. Requirement gathering** 12](#_Toc3874106)

[**2.3.1. Requirement Gathering methodologies** 12](#_Toc3874107)

[**2.3.2. Result found** 12](#_Toc3874108)

[**2.3.3. Business Rules** 13](#_Toc3874109)

[**2.4. Proposed system** 13](#_Toc3874110)

[**2.4.1. Overview** 15](#_Toc3874111)

[**2.4.2. Functional requirements** 15](#_Toc3874112)

[**2.4.3. Non-functional requirement** 18](#_Toc3874113)

[**2.5. System model** 20](#_Toc3874114)

[**2.5.1. Scenario** 20](#_Toc3874115)

[**2.5.2. Use case models** 34](#_Toc3874116)

[**2.5.3. Activity diagram** 60](#_Toc3874117)

[**2.5.4. Object model** 64](#_Toc3874118)

[**2.5.5. Dynamic modeling** 68](#_Toc3874119)

[**Chapter 3** 74](#_Toc3874120)

[**3. System design** 74](#_Toc3874121)

[**3.1. Introduction** 74](#_Toc3874122)

[**3.2. Current software architecture** 75](#_Toc3874123)

[**3.3. Proposed software architecture** 75](#_Toc3874124)

[**3.3.1. Overview** 75](#_Toc3874125)

[**3.3.2. Subsystem decomposition** 76](#_Toc3874126)

[**3.3.3. Hardware and software mapping** 77](#_Toc3874127)

[**3.3.4. Persistent data management** 78](#_Toc3874128)

[**3.3.5. Access control and security** 80](#_Toc3874129)

[**3.3.6 Subsystem service** 81](#_Toc3874130)

[**3.4. Detail Design** 82](#_Toc3874131)

[**3.4.1. Design class model** 82](#_Toc3874132)

[**3.4.2. Package** 84](#_Toc3874133)

[**CONCLUSION** 85](#_Toc3874134)

[**REFERENCE** 86](#_Toc3874135)

[**APPENDEX A** 87](#_Toc3874136)

# **LIST OF FIGURE**

[**Figure 2: Iterative development approach** 5](#_Toc2907023)

[**Figure 3: use case diagram for FYIPMS** 35](#_Toc2907024)

[**Figure 4: activity diagram for login** 60](#_Toc2907025)

[**Figure 5: activity diagram for assign group** 61](#_Toc2907026)

[**Figure 6: activity diagram for create account** 61](#_Toc2907027)

[**Figure 7: activity diagram for activate account** 62](#_Toc2907028)

[**Figure 8: activity diagram for add mark** 63](#_Toc2907029)

[**Figure 9: activity diagram for generate report** 63](#_Toc2907030)

[**Figure 10: activity diagram for submit proposal** 64](#_Toc2907031)

[**Figure 11: analysis class model** 67](#_Toc2907032)

[**Figure 12: state chart diagram for checking plagiarism** **Error! Bookmark not defined.**](file:///C:\Users\Kal\Desktop\fyipms%20last\FYIPMS%20print.docx#_Toc2907033)

[**Figure 13: sequential diagram for assign coordinator** 69](#_Toc2907034)

[**Figure 14: sequential diagram for assign group** 70](#_Toc2907035)

[**Figure 15: sequential diagram for submit proposal** 70](#_Toc2907036)

[**Figure 16: sequential diagram for checking plagiarism** 71](#_Toc2907037)

[**Figure 17: sequential diagram for approve completed document** 71](#_Toc2907038)

[**Figure 18: user interface for login** 72](#_Toc2907039)

[**Figure 19: user interface for advisor give comment** 73](#_Toc2907040)

[**Figure 20: three tire architecture** 76](#_Toc2907041)

[**Figure 21: sub system decomposition of FYIPMS by using component diagram** 77](#_Toc2907042)

[**Figure 22: deployment diagram for FYIPMS** 78](#_Toc2907043)

[**Figure 23: persistence data management for FYIPMS** **Error! Bookmark not defined.**](file:///C:\Users\Kal\Desktop\fyipms%20last\FYIPMS%20print.docx#_Toc2907044)

[**Figure 24: detail class diagram** 83](#_Toc2907045)

[**Figure 25: package diagram 84**](#_Toc2907046)

# **LISTOF TABLE**

[**Table 1: login use case description for users** 37](#_Toc2907140)

[**Table 2: create account use case description for system admin** 38](#_Toc2907141)

[**Table 3: activate account use case description for system admin** 38](#_Toc2907142)

[**Table 4: deactivate account use case description for system admin** 39](#_Toc2907143)

[**Table 5: assign group use case description for coordinator 40**](#_Toc2907144)

[**Table 6: manage rule use case description for coordinator 41**](#_Toc2907145)

[**Table 7: give comment use case description for advisor** 42](#_Toc2907146)

[**Table 8: view comment use case description for student** 42](#_Toc2907147)

[**Table 9: submit proposal with group use case description for student** 44](#_Toc2907148)

[**Table 10: send project use case description for student** 44](#_Toc2907149)

[**Table 11: check plagiarism use case description** 45](#_Toc2907150)

[**Table 12: assign advisor use case description for coordinator** 46](#_Toc2907151)

[**Table 13: assign examiner use case description for coordinator** 47](#_Toc2907152)

[**Ta+ble 14: add mark use case description for examiner and advisor** 48](#_Toc2907153)

[**Table 15: view mark use case description for student and coordinator** 49](#_Toc2907154)

[**Table 16: post notice use case description for coordinator** 49](#_Toc2907155)

[**Table 17: view notice use case description for student and examiner** 50](#_Toc2907156)

[**Table 18: approve completed document use case description for advisor** 51](#_Toc2907157)

[**Table 19: generate report use case description for coordinator** 52](#_Toc2907158)

[**Table 20: view generate report use case description for department head** 52](#_Toc2907159)

[**Table 21: update account use case description for users** 54](#_Toc2907160)

[**Table 22: upload material use case description for coordinator** 55](#_Toc2907161)

[**Table 23: download material use case description for student** 55](#_Toc2907162)

[**Table 24: assign coordinator use case description for department head** 56](#_Toc2907163)

[**Table 25: send progress use case description for advisor** 57](#_Toc2907164)

[**Table 26: view progress use case description for coordinator use case** 57](#_Toc2907165)

[**Table 27: view feedback use case description for coordinator** 58](#_Toc2907166)

[**Table 28: advisor send approve document** 59](#_Toc2907167)

[**Table 29: student send feedback** 59](#_Toc2907168)

[**Table 30: data dictionary** 65](#_Toc2907169)

[**Table 31: access control** 81](#_Toc2907170)

# **ABSTRUCT**

This documentation explains about web based Final Year Industrial Project Management System for University of Gondar Faculty of Informatics department of computer science. Currently the services that are given in Final Year Industrial project management system is handled manually. So to give service and manage difficult Final year Industrial Project Management System. It contains the introduction and background of the organization, the problem of statement, the objective and significance of the project in chapter one. In chapter two it contains the requirement analysis activities such as current system description and proposed system analysis such as scenario; use case diagram, use case description, activity diagram, sequence diagram, class modeling, and user interface. In chapter three we describe system design activities such as current software architecture, proposed software architecture, subsystem decomposition hardware/software mapping, persistent data management strategy access control and security, subsystem service, detailed class diagram and packages.

**KEYWORD:**

**CHAPTER ONE**

# **Introduction**

## **Background of the study**

Nowadays, the growing of computer technology is too fast, this changes the way we do business, the way we live, the way we communicate, and others. From this, most universities around the globe are using computer technology to improve their services and satisfy their customers’ interests. So, the computer technology has made a big difference in the teaching process. In our country there are 46 universities that have been in service. From these, University of Gondar is one of the governmental higher universities located in Gondar, Ethiopia. The establishment of university of Gondar date back to 1954 G.C as Public Health College and training center (PHC&TC) was founded In 1980 G.C. The training center was renamed Gondar College of medical science (GCMS) In 2003 G.C. The name changed from GCMS to Gondar University College. The university gets its current name in 2004 G.C. Hence, Gondar College changes to University of Gondar. The university has five main campuses, namely Atse Tewodros, Atse Fasil, Maraki, Medical Campus and Tseda campus. From these, Atse Tewodros campus has three colleges these are: Faculty of informatics, collage veterinary medicine and college of natural and computational science. The faculty of informatics was established in 2008 E.C after the college of medical science changed to the level of university [1]. The faculty of informatics has four departments, namely department of computer science, department of information system, department of information science, department of information technology from these faculties; our project has been concerned on faculty of informatics department of computer science. Currently, Gondar University has been using various systems to facilitate the teaching and learning mechanisms; among these systems, online registration system, portal for exchanging information, academic resource and online examination systems and portal for researching community service and technology transfer. But it has no undergraduate final year project management system. Final year project is a compulsory subject for every university students. As UOG, Final Year Project System (FYPS) does not have any system to purely deal and manage Final Year Project System (FYPS) related works. For this matter, it takes a lot of man power to do manual works. The process of coordinating group of students to prepare projects either to complete a single course curriculum or to graduate is expected to do a lot of works. Helping the students to submit title for their project, they should lay their hands on already completed projects to assign their advisors so as to be helped with their project. Assigning examiner to evaluate each finished project, setting schedule of the project, controlling each step of every process, and preserving the completed project are some of the activities that are done manually to complete the process of coordination.

This project emphasizes on the manual system that is described above. After studying this manual system by using different types of methodologies, we have developed this proposed project to simplify the coordinating system in to computerized system. Generally, this system has major mechanisms to submit proposals online, approve or reject proposal based on the content of the proposed project online, department head assign coordinator, coordinators assign advisor and examiner, post notice online, enable students and advisors to contact online in addition to be face to face.

In this paper, we present an innovative final year industrial project management system (FYIPMS) to keep track of different project starting from selecting projects going through until evaluation.

## **1.2. Statement of the problem and justification**

Currently, department of computer science uses manual approach, this means is that everything related to FYP works are done manually without technology based approach. Still now the department has no any computerized system to manage the FYP and related works.

We have analyzed the existing system from different point of view and stumbled upon the following problem:

* Department head assign coordinators, coordinators assign group of students, assign advisors based on their interest, assign examiners and also students submit their group and proposal manually these results time wastage and stress full long activity.
* Students are allocated to different advisors a project carried out by students in particular year with certain advisor could be picked up by another student in another year and replicate to another advisor with same department without the advisors knowledge this cause so many duplication of projects.
* Advisors and examiners add mark manually, projects or documents are stored in physically result these leads to mark papers and project documents lost easily and it is difficult to use previous projects for reference.
* It is difficult to control students and advisors activity these results either the student or the advisor could be sluggish with their responsibility.
* It cannot manage rule at a time so students confuse the work and coordinators cannot generate report on time.
* Posted notice lost from the notice bored.

## **1.3. Objective of the study**

Generally, our system has two objectives. These are: General objective and Specific objectives.

### **1.3.1. General objective**

The main objective of this project is to develop web based FYIPMS for department of computer science.

### **1.3.2. Specific objective**

In order to achieve the general objective, the specific objectives have the following:-

* Understand the existing system.
* Gather required information for the proposed system.
* Design and Develop a new system.
* Implement the new system.

## **1.4. Scope of the project**

This project emphasizes on developing web based FYIPMS for university of Gondar faculty of informatics department of computer.

## **1.5. Limitation of the project**

This project is limited only to those activates and operations related to FYP which the team planned to deal with and the challenges that will limit our work. This project is limited to develop the web based FYIPMS. Our project has the following limitation:

* After the new proposed system is implemented, the infrastructure like Internet connection will affect our system.
* In our proposed project we do not include grading.

## **1.6. System development methodology**

* **System analysis: I**s the process of breakdown an entire system into module, analyzing each module separately, and determining the relationship between them. In system development process we use system modeling i.e. creating model of the system. System modeling is the process of creating a model of system by analyzing and organizing the system element. This is the phase where deeply understanding of the existing system problem and finding alternative solution.
* **System design:** To design the system, the project team has chosen objective oriented modeling techniques and unified modeling language tools.
* **Development approach**: The development approach model for this system project selects iterative approach. The process of development in cyclic manner by repeating every step after every cycle of system development life cycle (SDLC) process. Iterative approach used to encourage innovation, increase flexible design, and provide quick implementation of an incomplete, but functional and application. It helps to easily identify confusing or difficult functions and missing functionality. The spiral model challenging to determine the exact composition of development methodologies to use each iteration around the spiral model [2].

**Requirement gathering**

**Figure 1: Iterative development approach**

#### **Data gathering methods**

We are going to use different types of data gathering methods for developing our system. This are:

* **Observation**
* **Interview**
* **Literature review**

#### **System development tool**

The web based application developed by using HTML, JavaScript, cascading style sheet (CSS) as the front end and PHP, MySQL as the back end.

* **PHP** is a server-side scripting language designed primarily for web development but also used as general-purpose programing language.
* **HTML** (Hypertext Markup Language) is the standard markup language for creating web page and web application.
* **CSS** is the language for describing the presentation of web pages, including colors, layout, and fonts. it allow one to adapt the presentation to different type of devices, such as large screens, small screens, or printers.
* **JavaScript** is supported by all modern web browsers and is used on almost every site on the web for more powerful and complex functionality. In short, JavaScript is a programming language that lets web developers design interactive sites.
* **MySQL** is an open source relational database management system. It runs as a server and allows multiple users to manage and create numerous database. It is central component in the WAMP stack of open source web application software that is used to create websites. WAMP stands for Window, Apache, MySQL, and PHP.

##### **Software to be used**

* **Microsoft word:** used for documentation.
* **Microsoft Visio, edrawmax and visual paradigm:** This is used to draw different UML (unified modeling language) that are necessary to structure the system.
* **Microsoft PowerPoint:** is software that we will use for presentation.
* **Notepad++:** to write PHP code.
* **Server side scripting**: hypertext pre-processor (PHP), we have selected PHP for server side scripting because it has the following advantages; PHP runs on different platforms (windows, Linux, UNIX, etc.), PHP is compatible with almost all servers used today apache and PHP is easy to learn and runs efficiently on the server side.

##### **Hardware to be used**

* Any computer (personal computer and desktop)
* 4GB RAM.
* Printer: for printing a document

## **1.7. Feasibility study**

Feasibility study is essential to evaluate the cost and benefits of the new system. On the basis of the feasibility study decision is taken on whether to proceed or to cancel the project**.**

We need the feasibility study:

* It determines the potential of the existing system.
* It used to determine/finds out the problem of the existing system.
* To determine all goals of the new system.
* It finds all possible solutions of the problems of the existing system.

### **1.7.1. Technical feasibility**

Currently the most web server support PHP. So we can get web server easily. The database server is MYSQL and it is easily maintainable. Since all features are available on web the client don’t need to install other applications rather than a web browser. Unified modeling language (UML) model to do analyzing and designing in good manner. So the system will be technically feasible.

### **Economic feasibility**

This project is economically feasible. It may save the resources like paper and other stationary materials, time and on the other side the man power usage in FYIPMS will decreases. So it is economically feasible.

### **Operational feasibility**

The client can get a user friendly GUI and easily operable system. So the system is operational feasible. It determines how the proposed system will satisfy the organizations need and it also offer secure, accurate and efficient system to the organization. The system in which we are developing is also compatible to all operating system and web browsers. So the proposed system is operational feasible. Those users have somewhat computer knowledge’s hence can understand and use this system. Therefore, the system will be designed to be operationally feasible. In addition the system is practical and applicable.

## **Significance of the project**

FYIPMS provides a web based easy-to-use, flexible solution to manage these task of all stakeholders. In terms of system functionalities and interactions with the system.

There are many benefits gained upon this project’s completion some are stated as follow:

* There will be a checker which used to check the similarity between two project proposals this used to students do not copy others work is no duplication of projects.
* Coordinators assign group, students submit their project proposal and group, Assign advisors and examiner for each group online and also Department head can assign coordinators and view report on time.
* There will be online comment giving and accepting mechanism.
* Advisor or students don’t need schedule every meeting.
* Students can send feedback to the coordinator,
* Project or documents are stored in database that stores useful project electronically this used to projects are permanently stored and easy to use previous projects for reference.
* It helps to communicate and crate healthy relationships with the stakeholders through the FYIPM process.

## **Beneficiaries**

There are also different bodies that will be benefited from this system. Among those some of them are:

* **For the department**: - once the new system is implemented the quality and performance of their work is improved.
* **For the Coordinators**: - once the new system is implemented the work load of coordinators will reduce. Selecting project title, Assigns advisors and examiners for each groups online, so this results time saving and title is understanding.
* **For the Advisors and students** :- once the new system is implemented the advisor and student do not expect to meet face to face to schedule every meeting there will be an online comment giving and accepting mechanism.

## **Project organization**

The project document is divided into five chapters for better understanding these are:-

* **Chapter one** focuses on introductory part as we have seen above.
* **Chapter two** of the document is all about the feature of the system including description of existing and proposed system, requirements like functional and nonfunctional requirements, the analysis models like use cases, state chart, sequence, activity diagram, analysis class model.
* **Chapter three** on this chapter we discussed what the system design should look the current software architecture (if any), proposed software architecture and detailed class diagram.
* **Chapter four** the project document is about the system implementation & coding part of the project. It clearly shows the interface implementation testing part of the project.
* **Chapter five** on this chapter we discuss conclusion and recommendation of the new system.

## **Time schedule**



# **CAPTER TWO**

# **Requirement analysis**

## **2.1. Introduction**

Requirement analysis is a software engineering task that bridge the gap between system level requirement engineering and software design. Requirement analysis provides the software design with representation of information, function, and behavior that can be translated to data, architectural, interface, and component level design.

Detailing System Requirements is the initial step of constructing the whole software system. It is a description of the project requirements that we have been gather and analyze. Without requirements, the goal is unclear and the construction may be off track. In this chapter, requirements of the software system are defined through human natural language and use case. The document is provided in order to ensure that the software we produce will be consistent with the need of the system user. Stating these requirements explicitly helps insure that any potential miscommunications are corrected at early stage.

The aim of this document is to gather, analyze and give an in-depth insight of the complete **Web Based FYIPMS** by defining the current system problem statement in detail. Nevertheless, it also concentrates on the functional and nonfunctional requirements of the system.

## **2.2. Current system description**

The current system of FYIPMS of department of computer science is manual to keep the data in store. The system didn’t use any office automation infrastructures to provide services for the university. Following such system is time consuming, tedious and it also consumes several resources and manpower. This is because until now there is no automated system developed for the department.

### **2.2.1. Major functions of the current system**

The existing system has the following functionality manually:

* Coordinator assign groups based on their GPA, student submit their group with titles orderly in manual way.
* Coordinator analyzes the submitted titles and assigns one title for each group of students.
* Coordinator assigns each group of students with advisor manually.
* Department head assign coordinators manually and view the report at the end of the year.
* Students and advisor physically meet occasionally so that the students could show the advisor their project status and the advisor could give comment and suggestion in return manually.
* Upon project completion the students submit their project manually.
* After submission of final projects examiners or evaluators will be assigned for each group by the project coordinator.
* Coordinator assign evaluation date will be notice and posted on the notice board manually.
* On the scheduled day of evaluation projects will be evaluated and examiners will give comment and submit grades of the projects they evaluated manually.
* Collect mark from advisor and examiner manually.
* Cannot manage rule at a time so the student can confused the work.
* Coordinator generate annually report.

#### **Problem of the current system**

As explained above currently FYIPMS that the department of computer science using is manual system for grouping students ,selecting project title, assigning advisors, and other activities which is done manually. So there is some drawback as we try to observe and know about the current system works.

* **Time consuming**: - since the FYP grouping students, selecting title and other related works process done by human power it may require more time for analyzing and comparing the status.
* **Lack of reliability and prone to error**: -since the FYP related works done by humans then they may conduct error
* **Need more human power**: - as the work need more care and focus it is difficult to perform with less human power.

## **2.3. Requirement gathering**

This section describes the data collection methods that we use to solve the problem and how the information is gather.

### **2.3.1. Requirement Gathering methodologies**

#### **Observation:** observing previous project. We have tried to observe the project advising, examination method and other FYP related works and tries to understand the manual system.

* **Interview:** To gather information from students, advisors, examiners, coordinators. We have interview coordinators and students gather some information that is vital to our system requirement.
* **Literature review: -**review all necessary documents, books that support this project. To make this project more feasible and structured we have tried to analyze other final year project management websites from universities in our country and others.

### **2.3.2. Result found**

Result found while gathering requirements regarding the problem of the current manual system in the department of computer science.

As we interview the coordinators and the students they said that

* Due to the manual system is tedious to assign group it is highly prone to error because when coordinators assign the group regarding to students GPA.
* Since the project title selecting process is leads to duplication of projects.
* It is difficult to control students and advisor activities this result either the students or the advisors could be sluggish with their responsibility.
* Projects or documents are stored in physically results so documents are lost easily.
* Accepted title and schedule are posted on the notice board so the posted schedule are gone be lost or tear by some unethical student.

These result used as a starting point to the main functionality of the new system.

### **2.3.3. Business Rules**

Any organization has its own rule and regulation to run the system in good manner. A business rule is effectively an operating principles or policy that software must satisfy and it is what a guideline that are fulfilled by the functional requirement. Identifying and documenting business rules are thus important because it enable us to refer easily rules for other development artifacts, such as model and use case. These business rules of department of computer science FYIPMS are described as follows.

* Unauthorized user cannot access the system for the reason of system security.
* Every user of the system must have an account for accessing the system, even the administrator itself to increase the security of the system.
* Every user of the system must be a member of the department.
* In our proposed project the department head can’t be coordinator, advisor, examiner, also the coordinator can’t be advisor or examiner.
* The plagiarism detector will evaluate project proposal below 20% would probably indicate that plagiarism has not occurred.

## **2.4. Proposed system**

The main goals of the proposed system are designing and constructing a new web based system for department of computer science. Since it make us the feasible option in addressing both the problems of the existing system by giving additional freedom to administrate, to maintain and upgrade the system.

As we have tried to mention in the previous chapter, the need to develop web based FYIPMS for department of computer science is that the current activities are time consuming due to manual system. The newly proposed system is efficient in facilitating the different tasks.

The proposed system can facilitate the following activity: -

* Department head assign coordinator.
* Coordinators assign group online.
* Students submit proposal with group online.
* There will be a checker which used to check the similarity between project proposals and evaluate (accept or reject) online.
* Students can view accept title, view mark, download material, view comment, view notice online.
* Coordinators assign advisors and examiner.
* Coordinators can post notice for proposal and final document presentation.
* Document (project) resources are stored electronically and are easily available for next users as reference.
* Students and advisor contact online which used to students submit their work online for comment and the project couldn’t be lost by anyway.
* Examiners can view notice.
* Examiners return evaluated project after evaluating and rating to coordinator.
* Coordinators to collect mark from advisor and examiner easily through the system.
* Coordinators manage rule at a time the student cannot confuse the work.
* Coordinators can generate annually report.
* Department head can view annually report on time.

Our proposed system is better than the existing system because of the following advantages

* **Little time consuming**:-since the grouping, title selecting, assigning advisor, assigning examiner and others processes are done by computer it may not require more time than the existing system.
* **No duplication or redundancy**: - duplication occurs when some students lay there hand on already done projects. Since the system detect by computerized system the probability occurrence of duplicated projects exactly zero.
* **More reliable** :- since the FYP related works done by computerized system then it does not conduct error due to stress, occurrence of boring on the work and such reasons. So the system become more reliable.
* **Flexible**: - since the advisors and students can contact online in steady of face to face, coordinators can generate annual report on time.
* **It reduces work load**: - since the system work through the help of computer, a lot of work are done with in the fraction of seconds.

### **2.4.1. Overview**

This section of the document provides a general description including high level functionality of the system and the systems non-functional requirements defined in terms of human natural language and use cases.

### **2.4.2. Functional requirements**

Functional requirement are the main things that the user expect from the system. It describes what the system should do and behavior of the system as it relates to the system functionality. It also describes the relation between the system and the user or the environment. Her the ‘relation’ means the direct or indirect interaction between the system and the user.

**STUDENT:**

* **Submit proposal with group**: In order to continue the project work students able to submit title and proposal with group by using the system.
* **Send project:** In order to approve the completed document by the advisor and to get to the evaluation the students able to send the project by using the system.
* **Check plagiarism:** In order to evaluate student’s project proposal first the students able to upload the proposal for the checker by using the system and the checker accept or reject the proposed project.
* **View notice:** In order to know about the presentation date and other project work dead line the student able to view the notice by using the system.
* **Send Feedback:** the system shall allow the students to send complain by using the system.
* **View comment**: The system shall allow students to view comment which is send by the advisor and examiner by using the system.
* **View assessment:** The system shall allow student to view their assessment which is the advisor and examiner give assessment to the system and student can view their assessment by using the system.
* **Download material:** The system shall allow students to download sample documents and guidelines.

**ADVISOR:**

* **Give comment:** The system shall allow advisor to give comment for the student by using the system.
* **Approve completed document:** In order to the student get to examination the advisor should approve the completed document by using the system.
* **Give assessment:** The system shall allow the advisor to add mark to student by using the system.
* **Send approve document:** The system shall allow advisor to send approve document by using the system for the examiner.
* **Send stages:** The system shall allow advisors to send stages/progress by using the system for the coordinator.

**COORDINATOR:**

* **View feedback:** in order to view students complain the system shall allow the coordinator to view complain by using the system.
* **Assign students:** Coordinators able to assign group of students by using the system.
* **Assign advisor:** The system shall allow coordinators to assign advisors for each group by using the system.
* **Assign examiner:** The system shall allow coordinators to assign examiners by using the system for examining student project.
* **Post notice:** In order to know if there is any new thing or when will be the presentation date and other information. The coordinator should be able to post notice by using the system.
* **Upload material:** The system shall allow the coordinator to upload materials by using the system.
* **View stages:** In order to follow progress of student and advisor activity the system shall allow coordinator to view progress by using the system.
* **Mange rule:** in order to students work properly the system shall allow coordinator to manage rule.

**EXAMINER:**

* **Add mark**: The system shall allow examiners to add evaluated mark to students by using the system.
* **View notice**: The system shall allow examiners to view notice /schedules by using the system.
* **View approve document**: the system allow the examiner to view approve document by using the system.
* **Give comment:** The system shall allow examiners to give comment for the students by using the system.

**DEPARTMENT HADE**

* **Views generate report:** The system shall allow the department head to view generated report by using the system.
* **Assign coordinator:** The system shall allow department head to assign coordinator by using the system.

**SYSTEM ADMIN:**

* **Crate account:** The system shall allow system admin to create users account.
* **Activate account:** The system shall allow system admin to activate users account by using the system.
* **Deactivate account:** The system shall allow system admin to deactivate users account by using the system.
* **Update account:** The system shall allow system admin to update users account by using the system.

### **2.4.3. Non-functional requirement**

#### **2.4.3.1. User interface and human factor**

The expert that is required from user is only knowledge of computer usage and web access. The interface of the proposed system is very flexible for users. The system uses English language. The user of this system must have knowledge of how about computer.

#### **2.4.3.2. Documentation**

The documentation should only contain user manual to provide a small help guide and tool tips.

#### **2.4.3.3. Hardware consideration**

The Hardware required to use the system is basically a computer machine that is used one as a client and the other as a server. The computer machine can be a laptop or desktop computers used for accessing the system and the server is used for storing the website and receiving and responding response.

#### **2.4.3.4. Performance characteristics**

The proposed system must have 1 second response time and 2 second response time in worst case. The system can also support concurrent 300 users simultaneously this is with best response time. The system must support parallel transactions involving different clients from different location at a time.

* **Response time**: the output should be generate with maximum of 4 second depending on the internet connection speed.
* **Capacity**: the system should support 300 concurrent users.
* **Resource utilization:** the application should utilize minimum amount of CPU and memory of the device.

#### **2.4.3.5. Error handling and extreme conditions**

If the users enter incorrect input like inserting empty string, inserting duplicated username in to the database or login with unregistered username and password the system should display an appropriate message for each error. The system should handle exceptions like login error (unauthorized login), submit empty string, attempt to insert duplicate username into the database while registration, inserting alphabetic value in integer and vice versa.

#### **2.4.3.6. Quality issues**

* **Reliability:** The systems not fail in month.
* **Usability:** the system should be easy to learn and operate. It need only fraction of hour to introduce users to the system.
* **Availability:** the system should be available for all working hour.
* **Portability:** the system operates only on windows operating system.

#### **2.4.3.7. System modification**

As the world is capable of change time to time there will be future change to the system as result of new innovation therefor the system can be upgrade to the new technology by the maintainer of the system developers.

#### **2.4.3.8 Physical environment**

UI part of the system will be deployed on the client device and database part of the system will be deployed on central server located in tewodrose campus. While users will retrieve data from the central server.

#### **2.4.3.9. Security issues**

Security requirement represent the environment in which the system must operates as well as the type and degree of security that must be provide.

#### **2.4.3.10. Resource issues**

* Server: minimum hardware requirements for apache server

-core: single core 3 HZ or single dual 2GHZ

-display resolution: 1360x768(recommended high)

* Client: CPU: 32/64 bit

## **2.5. System model**

### **2.5.1. Scenario**

Scenario is a brief user story explaining who is using the system and what they are trying to accomplish. Based on this, scenarios related to the system functional requirements stated in use case description part.

**Scenario 1:**

**Scenario name: login**

**Participating actor: student, coordinator, advisor, examiner, department head, system admin**

**Flow events:**

1. The users wish to enter to the page.
2. The users click on the login button.
3. The system display the login form.
4. The user fill the user name and password.
5. The user clicks on the login button.
6. The system validate the user name and password.
7. If the user name and password correct the user are entered into the system based on their privilege.
8. The use case end.

**Alternative action:**

* If the users enter invalid user name and password, the system displays incorrect username and password. Go back to step 4.

**Scenario 2:**

**Scenario name: create account**

**Participating actor: system admin**

**Flow events:**

1. The system admin wishes to create an account.
2. The system admin selects create account link.
3. The system display create account page.
4. The system admin fill all the required field and submit it.
5. The system verify validation of inputs.
6. The system register the users in to the data base.
7. The system display user account create success fully.
8. Use case end.

**Alternative action:**

* If the information that the system admin is fill invalid information, the system displays duplicate entry of user name. Go back to step 2.

**Scenario 3**

**Scenario name: activate account**

**Participating actor: system admin**

**Flow event:**

1. The system admin wishes to activate users account.
2. The system admin selects accounts that to be activated.
3. The system display required accounts from the database.
4. The system admin clicks on activate button.
5. The system display activate account is successful.
6. Use case end.

**Scenario 4**

**Scenario name: deactivate account**

**Participating actor: system admin**

**Flow event:**

1. The system admin wishes to deactivate accounts.
2. The system admin selects account that to be deactivated.
3. The system display required accounts from the database.
4. The system admin clicks on deactivate button.
5. The system display deactivate account is successful.
6. Use case end.

**Scenario 5**

**Scenario name: submit proposal**

**Participating actor: student**

**Flow event:**

1. The student wishes to submit proposal with group.
2. The student clicks on the submit proposal link.
3. The system displays submit proposal form.
4. The student first enter the group and fill all the required filed orderly and click on the submit button.
5. The system validates the data entered by the student.
6. The student add the proposal title to title list.
7. The system display proposal submitted successfully message.
8. Use case end.

**Alternative action:**

* If the student entered incorrect information the form is not fill successfully, the system displays error message and go back to step 4.

**Scenario 6**

**Scenario name: assign group**

**Participating actor: coordinator**

**Flow event:**

1. The coordinator wishes to assign group.
2. The coordinator click on assign group link.
3. The system displays students list and a form how to group student page.
4. The coordinator fill the form and click on assign group button.
5. The system display group assign successfully message.
6. Use case end.

**Scenario 7**

**Scenario name: view comment**

**Participating actor: student**

**Flow event:**

1. The student wishes to view comment.
2. The student click on view comment link.
3. The system displays comments.
4. Use case end.

**Scenario 9**

**Scenario name: assign advisor**

**Participating actor: coordinator**

**Flow event:**

1. The coordinator wishes to assign advisor.
2. The coordinator click on assign advisor link.
3. The system displays assign advisor page.
4. The coordinator selects advisors from department list.
5. The system displays list of title in department.
6. The coordinator selects advisor for titles.
7. The coordinator click on assign advisor button.
8. The system displays assign advisor successfully message.
9. Use case end.

**Scenario 10**

**Scenario name: assign examiner**

**Participating actor: coordinator**

**Flow event:**

1. The coordinator wishes to assign examiner.
2. The coordinator click on assign examiner link.
3. The system displays assign examiner page.
4. The coordinator selects examiner form department list.
5. The system displays list of title in department.
6. The coordinator selects examiner for titles.
7. The coordinator click on assign examiner button.
8. The system displays assign examiner successfully message.
9. Use case end.

**Scenario 11**

**Scenario name: generate report**

**Participating actor: coordinator**

**Flow event:**

1. The coordinator wishes to generate report.
2. The coordinator click on generate report link.
3. The system displays generate report page.
4. The coordinator fill all the necessary information and click on send report button.
5. The system check all the filled information and displays successfully message.
6. Use case end.

**Alternative action**

* If the form is not filled correctly go back to step 4.

**Scenario 12**

**Scenario name: approve completed document**

**Participating actor: advisor**

**Flow event:**

1. The advisor wishes to approve completed documents.
2. The advisor select completed document.
3. The system display completed document.
4. The advisor click on approve button.
5. The system displays successful message.
6. Use case end.

**Scenario 13**

**Scenario name: send approve document**

**Participating actor: advisor**

**Flow event:**

1. The advisor wishes to send approve documents to the examiner.
2. The advisor login to the system.
3. The advisor click on send approved document link.
4. The system display approved documents with examiners list.
5. The advisor clicks on send approved document button.
6. Use case end.

**Scenario 14**

**Scenario name: give assessment**

**Participating actor: advisor, examiner**

**Flow event:**

1. The users wishes to give assessment.
2. The users click to give assessment link.
3. The system displays give assessment page.
4. The users fill appropriate information on the form.
5. The users click on the add assessment button.
6. The system check all filled information.
7. The data save to database.
8. Use case end.

**Alternative action:**

* If the form is not filled correctly, the system display error message and back to step 4, 5 and 6.

**Scenario 15**

**Scenario name: post notice**

**Participating actor: coordinator**

**Flow event:**

1. The coordinator wishes to post notice.
2. The coordinator click on post notice link.
3. The system displays post notice page.
4. The coordinator writes post information that include presentation date.
5. The coordinator click on post notice button.
6. The system displays notice post successfully.
7. Use case end.

**Scenario 16**

**Scenario name: view notice**

**Participating actor: student, examiner**

**Flow event:**

1. The users wishes to view posted notice.
2. The users click on view notice link.
3. The system displays view notice page.
4. The users view notice.
5. Use case end.

**Scenario 17**

**Scenario name: update account**

**Participating actor: system admin.**

**Flow event:**

1. The system admin wishes to update users account.
2. The system admin click on update account link.
3. The system displays update account page.
4. The system admin select information to be update.
5. The system admin click update button.
6. The system display the modified information.
7. The system admin click save the modified information.
8. The system displays update successfully.
9. Use case end.

**Scenario 18**

**Scenario name: view assessment**

**Participating actor: student, coordinators**

**Flow event:**

1. The users wishes to view the added assessment.
2. The users click on view assessment link.
3. The system displays view assessment page.
4. The users view the assessment.
5. Use case end.

**Scenario 19**

**Scenario name: send project**

**Participating actor: student**

**Flow event:**

1. The student wishes to send project.
2. The student click on send project link.
3. The system display send project page.
4. The student upload the project.
5. The student click on send project button.
6. The system display project send successfully message.
7. Use case end.

**Scenario 20**

**Scenario name: give comment**

**Participating actor: advisor**

**Flow event:**

1. The user wishes to give comment.
2. The user click on give comment link.
3. The system displays give comment page.
4. The user fill all the necessary information in the form and writes comment and suggestion.
5. The user click on send comment button.
6. The system display comment send successfully message.
7. Use case end.

**Alternative course of action**

* If the users enters incorrect information (formatting like email, phone) or the form is not filed successfully, the system displays error message and go back to step 4.

**Scenario 21**

**Scenario name: upload comment**

**Participating actor: examiner**

**Flow event:**

1. The examiner wishes to upload comment of the documentation for students.
2. The examiner login to the system.
3. The examiner click on upload comment link.
4. The examiner fill all the necessary information in the form and upload the comment.
5. The examiner click on upload comment button.
6. The system display comment send successfully message.
7. Use case end.

**Scenario 22**

**Scenario name: manage rule**

**Participating actor: coordinator**

**Flow event:**

1. The coordinator wishes to manage rule.
2. The coordinator click on manage rule button.
3. The system displays manage rule page.
4. The coordinator fill all the necessary information step by step.
5. The coordinator click on manage rule button.
6. The system validate all the entered information and save it to the database.
7. The system displays successfully message.
8. Use case end.

**Alternative course of action**

* If the form is not filled properly the system displays an error message to fill the form correctly. And back to step4.

**Scenario 23**

**Scenario name: upload material**

**Participating actor: coordinator**

**Flow event:**

1. The coordinator wishes to upload material.
2. The coordinator click on document link.
3. The system display document page.
4. The coordinator click on upload document link.
5. The system display file browsing window.
6. The coordinator click on browse button.
7. The coordinator selects file from local computer.
8. The coordinator click on upload button.
9. The document upload to the server.
10. Use case end.

**Scenario 24**

**Scenario name: download material**

**Participating actor: student**

**Flow event:**

1. The student wishes to download material.
2. The student click on download sample document button.
3. The system displays sample document alternatives.
4. The system process the download that click the student.
5. Download finish.
6. Use case end.

**Scenario 25**

**Scenario name: assign coordinator**

**Participating actor: department head**

**Flow event:**

1. The department head wishes to assign coordinators.
2. The department head click on assign coordinator link.
3. The system display assign coordinator page.
4. The department head select coordinator from department list.
5. The system displays list of coordinators.
6. The department head selects coordinators.
7. The coordinator click on assign coordinators button.
8. The system displays assign coordinator successfully message.
9. Use case end.

**Scenario 26**

**Scenario name: view generate report**

**Participating actor: department head**

**Flow event:**

1. The department head wishes to view report.
2. The department head click on view report link.
3. The system display generated report page.
4. The department head view report.
5. Use case end.

**Scenario 27**

**Scenario name: send stages/progress**

**Participating actor: advisor**

**Flow event:**

1. The advisor wishes to send stages/progress follow up to the coordinator.
2. The advisor login to the system.
3. The advisor click on send stages/progress link.
4. The system display send stages/progress page with its form.
5. The advisors fill all the necessary information on the form.
6. The advisor click on send stages/progress button.
7. The system display successfully stages/progress send massage.

**Alternative course of action**

* If the advisor did not fill the form correctly the system display an error massage and back to step 5.

**Scenario 28**

**Scenario name: view stages/progress**

**Participating actor: coordinator**

**Flow event:**

1. The coordinator wishes to view stages/progress.
2. The coordinator login to the system.
3. The coordinator click on view stages/progress link.
4. The system displays view stages/progress page.
5. The coordinator view stages/progress.
6. Use case end.

**Scenario 29**

**Scenario name: check plagiarism**

**Participating actor: student**

**Flow event:**

1. The student wishes to check the content of the project proposal.
2. The student login to the system.
3. The student click on check plagiarism link.
4. The system display check plagiarism page.
5. The student upload the file to the checker.
6. The checker check the proposal and evaluate the proposal display the result to the student.
7. Use case end.

**Scenario 30**

**Scenario name: log out**

**Participating actor: student, coordinator, advisor, examiner, department head, system admin.**

**Flow event:**

1. The user wishes to log out from the system.
2. The users click on logout button.
3. The system displays main page.
4. Use case end.

### **2.5.2. Use case models**

The main activities that are performed in this sections are

* Identify if there is any additional actors and use case.
* Construct a use case model and
* Documenting the use case course of events.

#### **2.5.2.1. Use case diagram**

A use case diagram is a representation of a user’s interaction with the system that shows the relationship between the users and different type of users of a system and the different use case [3].



**Figure 2: use case diagram for FYIPMS**

#### **2.5.2.2. Use case description**

|  |  |  |
| --- | --- | --- |
| Use case name | Login | |
| Use case id | UC 01 | |
| Description | The user entered authorized username and password in order to access the system. | |
| Participating actor | Student, coordinator, advisor, examiner, system admin, department head. | |
| Precondition | The user should have valid username and password. | |
| Flow action | Actor action | System response |
| 1. The user Open home page.  3. Enter the authorized username and password in the login form.  4. Click on “login” button.  8. End use case. | 2. Display the login page with login form.  5. Check the authorized username and password.  6. If the user name and password is correct users are entered into system based on their privilege. |
| Alternative course of action | 7. If the username and password is incorrect back to step 3, 4 and 5 basic course of action. | |
| Post condition | The users successfully pass the login and do what they want to do in the system. | |

**Table 1: login use case description for users**

|  |  |  |
| --- | --- | --- |
| Use case name | Create account | |
| Use case id | UC 02 | |
| Description | The system admin creates to him and all other users in order to control system and users. | |
| Participating actor | System admin | |
| Per condition | The system admin login to the system and know all the users to create account and users should be member of the department. | |
| Flow of action | Actor action | System response |
| 1. The system admin login to the system.  2. The admin select create account link.  4. The administrator fills the required information and submit it.  9. Use case end. | 3. The system displayCreate account page.  5. The system verifies validation of inputs.  6. The system register the users into the database.  8. Display user account successfully created. |
| Alternative course of action | 7. If the information admin fill is invalid information. The system display duplicate entry of user name. go to step4 | |
| Post condition | The user can get user name and password to access the system. | |

**Table 2: create account use case description for system admin**

|  |  |  |
| --- | --- | --- |
| Use case name | Activate account | |
| Use case id | UC 02 | |
| Description | Used to activate users account | |
| Participating actor | System admin | |
| Per condition | The user information should be in the database. | |
| Flow of action | Actor action | System response |
| 1. The system admin login to the system.  2. The system admin selects account to be activated.  4. The administrator clicks activate button.  6. Use case end. | 3. The system display required account.  5. System displays activate account is successful. |
| Post condition | Activate account is successful. | |

**Table 3: activate account use case description for system admin**

|  |  |  |
| --- | --- | --- |
| Use case name | Deactivate account | |
| Use case id | UC 03 | |
| Description | Used to deactivate users account | |
| Participating actor | System admin | |
| Per condition | The user in formation should be in the database. | |
| Flow of action | Actor action | System response |
| 1. The system admin login to the system.  2. The admin select account that to be deactivate.  4. The administrator click deactivate button. | 3. The system displays required account from the database.  5. System display deactivate account is successful. |
| Post condition | Deactivate account successful. | |

**Table 4: deactivate account use case description for system admin**

|  |  |  |
| --- | --- | --- |
| Use case name | Assign group | |
| Use case id | UC 04 | |
| Description | Used to form group of students | |
| Participating actor | Coordinator | |
| Per condition | The coordinator must login to the page | |
| Flow of action | Actor action | System response |
| 1. The coordinator login to the system.  2. The coordinator click on assign group link.  4. The coordinator fill the form.  5. The coordinator click on assign group button. | 3. The system displays student list and assign group page.  6. The system check the filled information and displays group assign successfully. |
| Alternative course of action | 7. If the coordinator entered incorrect information (like how to group students) go back to step 4. | |
| Post condition | Students now there group | |

**Table 5: assign group use case description for coordinator**

|  |  |  |
| --- | --- | --- |
| Use case name | Manage rule | |
| Use case id | UC05 | |
| Description | The coordinator should manage rule for the improvement of work | |
| Participating actor | Coordinator | |
| Pre-condition | The coordinator must login to the system | |
| Flow of action | Actor action | System response |
| 1. The coordinator login to the system.  2. The coordinator click on manage rule link.  4. The coordinator fill all the necessary information step by step.  5. Clicks manage rule button. | 3. The system displays manage rule page.  6. The system validate all the entered information and save it to the database.  7. The system displays successfully message. |
| Alternative course of action | 6.1. If the form is not filled properly the system displays an error message to fill the form correctly. Back to step 4 | |
| Post condition | System display successful message. | |

**Table 6: manage rule use case description for coordinator**

|  |  |  |
| --- | --- | --- |
| Use case name | Give comment | |
| Use case id | UC 06 | |
| Description | The advisor write comment and suggestion and send to student. | |
| Participating actor | Advisor ,examiner | |
| Per condition | The advisor must login to the page. | |
| Flow of action | Actor action | System response |
| 1. The user login to the system.  2. The user click on the comment link.  4. The user fills the necessary information on the form and writes comment and suggestion.  5. The user click on send comment button.  7. Use case end. | 3. The system displays the comment form.  6. If the comment form is field successful the system display successfully message. |
| Alternative course of action | If the user enters incorrect information (formatting like email, phone) or the form is not field successfully, the system displays error message and go back to step 4. | |
| Post c6ondition | The comment is send successfully. | |

**Table 7: give comment use case description for advisor**

|  |  |  |
| --- | --- | --- |
| Use case name | View comment | |
| Use case id | UC 07 | |
| Description | The student view comment, suggestion which are send from the advisor. | |
| Participating actor | Student | |
| Per condition | The student must login to the system. | |
| Flow of action | Actor action | System response |
| 1. The student login to the system.  2. The student clicks on view comment link.  4. The student view comment. | 3. The system display view comment page. |
| Post condition | The student view comment. | |

**Table 8: view comment use case description for student**

|  |  |  |
| --- | --- | --- |
| Use case name | Submit proposal | |
| Use case id | UC 08 | |
| Description | The student submit their option of project proposal with group orderly to the project coordinator. | |
| Participating actor | Student | |
| Per condition | The student should login to system.  Student submit group first.  Students must submit their title and proposal. | |
| Flow of action | Actor action | System response |
| 1. The student first login to the system.  2. Students click submit proposal link.  4. Fill option of titles with proposal orderly.  6. Students click on submit button.  9. Use case end. | 3. The system display submit proposal form displayed.  5. System validates the data entered by students.  7. System adds title and proposal to title lists.  8. System displays proposal submitted successfully message. |
| Alternative course of action | 10. If the student enter incorrect information or the form is not field successfully, the system displays error message and go back to step 4. | |
| Post condition | Title will be activates for the project coordinator to approve. | |

**Table 9: submit proposal with group use case description for student**

|  |  |  |
| --- | --- | --- |
| Use case name | Send project | |
| Use case id | UC 9 | |
| Description | The student must send completed project to its advisor by using the system | |
| Participating actor | Student | |
| Pre-condition | The student must have valid user name and password. | |
| Flow of action | Actor action | System response |
| 1. The student login to the system.  2. The student click on send project link.  4. The student upload the project and click send project button.  6. Use case end. | 3. System display send project page.  5. Project send message displayed. |
| Post condition | Project send to the advisor | |

**Table 10: send project use case description for student**

|  |  |  |
| --- | --- | --- |
| Use case name | Check plagiarism | |
| Use case id | UC10 | |
| Description | The system should check the entered proposal and evaluate the proposal. | |
| Participating actor | Student | |
| Pre-condition | The student should have valid user name and password.  The student must enter the proposal for checking. | |
| Flow of action | Actor action | System response |
| 1. The student login to the system.  2. The student clicks on check plagiarism link.  4. The student upload the file to the system.  5. The student click check button. | 3. The system display checker page.  6. Result display to the coordinator. |
| Post condition | The entered proposal will be accepted or rejected by the coordinator. | |

**Table 11: check plagiarism use case description**

|  |  |  |
| --- | --- | --- |
| Use case name | Assign advisor | |
| Use case id | UC11 | |
| Description | The coordinator assign advisor | |
| Participating actor | Coordinator | |
| Per condition | The coordinator should login to the system. | |
| Flow of action | Actor action | System response |
| 1. Coordinators first must login to the system.  2. The coordinator clicks on assign advisors link.  4. Coordinators select advisors list from department list.  6. Select advisors for titles.  7. The coordinator clicks on assign button.  9. Use case end. | 3. The system displays assign advisor page.  5. Display list of titles in department.  8. assigned advisor successfully. |
| Post condition | Advisor are assigned to each group members. | |

**Table 12: assign advisor use case description for coordinator**

|  |  |  |
| --- | --- | --- |
| Use case name | Assign examiner | |
| Use case id | UC12 | |
| Description | The coordinator assign examiner. | |
| Participating actor | Coordinator | |
| Per condition | The coordinator must login to the system | |
| Flow of action | Actor action | System response |
| 1. The coordinator first login to the system.  2. The coordinator click on the assign examiner link.  4. The coordinator select examiner list from the department.  6. The coordinator click on assign button.  8. Use case end. | 3. The system display assign examiner page.  5. The system displays list of title from the department.  7. The system display assign examiner successfully. |
| Post condition | The coordinator will be assign examiner for each group | |

**Table 13: assign examiner use case description for coordinator**

|  |  |  |
| --- | --- | --- |
| Use case name | Add mark | |
| Use case id | UC13 | |
| Description | The examiner and advisor add mark of documentation and presentation | |
| Participating actor | Examiner and advisor | |
| Per condition | The examiner and advisor must login to the system first | |
| Flow of action | Actor action | System response |
| 1. The users login to the system.  2. The users click on the add mark link.  4.The user fill the appropriate information on the form.  5. The users click on add mark button.  9. Use case end. | 3. The system displays add mark page.  6. The system check all the filed information.  7. If the filled information is correct the data will be saved to the data base. |
| Alternative course of action | 1. If the entered data is in correct the system display error message and back to step 4, 5 and 6 of basic course of action. | |
| Post condition | The examiner and advisor add mark and send to coordinator and student. | |

**Table 14: add mark use case description for examiner and advisor**

|  |  |  |
| --- | --- | --- |
| Use case name | View assessment | |
| Use case id | UC 14 | |
| Description | The users able to view asessment which is add by the examiner, and advisor | |
| Participating actors | Student, coordinator | |
| Pre-condition | The users must login first | |
| Flow of event | Actor action | System response |
| 1. The users login to the system.  2. The users click on the view assessment link.  4. The users view the assessment.  5. Use case end. | 3. The system displays view assessment page. |
| Post condition | The users view the assessment. | |

**Table 15: view mark use case description for student and coordinator**

|  |  |  |
| --- | --- | --- |
| Use case name | Post notice | |
| Use case id | UC15 | |
| Description | Coordinators post notice including presentation date by using this system | |
| Participating actor | Coordinator | |
| Per condition | The coordinator must login to the system | |
| Flow of action | Actor action | System response |
| 1. The coordinator login to the system.  2. The coordinator click on post notice link.  4. The coordinator writes post information that includes presentation date.  5. Click on post notice button.  7. Use case end. | 3. The system displays post notice page.  6. The system displays notice post successfully. |
| Post condition | The student and examiner should get the posted notice/ or schedule. | |

**Table 16: post notice use case description for coordinator**

|  |  |  |
| --- | --- | --- |
| Use case name | View notice | |
| Use case id | UC 16 | |
| Description | The users must view notice which is posted by the coordinator | |
| Participating actor | Student, examiner | |
| Pre-condition | The users must login | |
| Flow of event | Actor action | System response |
| 1. The users login to the system.  2. The user’s clicks on view notice link.  4. The users view notice.  5. Use case end. | 3. The system displays view notice page. |
| Post condition | Users view notice. | |

**Table 17: view notice use case description for student and examiner**

|  |  |  |
| --- | --- | --- |
| Use case name | Approve completed document | |
| Use case id | UC17 | |
| Description | The advisor approve completed document to be examine. | |
| Participating actor | Advisor | |
| Per condition | The advisor must login | |
| Flow of action | Actor action | System response |
| 1. The advisor login to the system.  2. The advisor click on completed document link.  4. The advisor select completed document  5. Click on approve completed document button.  7. Use case end. | 3. The system display completed documentation list.  6. The system displays approve successfully message. |
| Post condition | The approved document must be examine | |

**Table 18: approve completed document use case description for advisor**

|  |  |  |
| --- | --- | --- |
| Use case name | Generate report | |
| Use case id | UC18 | |
| Description | The coordinator generate report about the overall work or activity. | |
| Participating actor | Coordinator | |
| Per condition | The coordinator must login | |
| Flow of action | Actor action | System response |
| 1. The coordinator login to the system.  2. The coordinator click on generate report link.  4. The coordinator review students and titles of this year and fill all the appropriate information  5. Click generate report button.  9. Use case end. | 3. The system displays generate report page with its form.  6. The system check the information filled on the form.  7. If the entered information is filled correctly the data saved to database. |
| Alternative course of action | If the form is not filled correctly the system displays error message and go back to step 4. | |
| Post condition | Report is generated successfully | |

**Table 19: generate report use case description for coordinator**

|  |  |  |
| --- | --- | --- |
| Use case name | View generated report | |
| Use case id | UC 20 | |
| Description | The department head view generated report form the coordinator | |
| Participating actor | Department head | |
| Per condition | The department head must login | |
| Flow of event | Actor action | System response |
| 1. The department head login to the system.  2. The department head click on view generated report link.  4. The department head view the report.  5. Use case end. | 3. The system display view generate report page. |
| Post condition | The department head view the report. | |

**Table 20: view generate report use case description for department head**

|  |  |  |
| --- | --- | --- |
| Use case name | Update account | |
| Use case id | UC21 | |
| Description | If there is any mistake during entering1 | |
| Participating actor | system admin | |
| Per condition | The system admin must create account for users (users should have user name and password). | |
| Flow of action | Actor action | System response |
| 1. The system admin login to the system.  2. The system admin click on update link.  4. The system admin select information to be update and click update button.  6. The system admin save the modified information.  8. Use case end. | 3. The system displays user information page.  5. The system displayed the modified information.  7. The system displays update successfully. |
| Alternative course of action | If the modified information is invalid the system display ‘please enter correct input’ message and return to basic course of action step 4 | |
| Post condition | Update successfully | |

**Table 21: update account use case description for users**

|  |  |  |
| --- | --- | --- |
| Use case name | Upload material | |
| Use case id | UC22 | |
| Description | The coordinator upload material | |
| Participating actor | Coordinator | |
| Pre-condition | The coordinator should first login to the system | |
| Flow action | Actors action | System respond |
| 1. Coordinator login to the system.  2. The coordinator click on document link.  4. The coordinator click on upload document link.  6. The coordinator click on browse button.  7. The coordinator select document from local computer.  8. The coordinator click on upload button.  10. Use case end. | 3. The system display document page.  5. The system display file browsing window.  9. Document uploaded to the server. |
| Post condition | The document is uploaded. | |

**Table 22: upload material use case description for coordinator**

|  |  |  |
| --- | --- | --- |
| Use case name | Download material | |
| Use case id | UC23 | |
| Description | The student can download sample document used as reference or guidelines. | |
| Participating actor | Student | |
| Post condition | The student must open the web page | |
| Flow action | Actor action | System response |
| 1. The student first login to the system.  2. The student click on download sample document button.  6. Usecase end. | 3. The system displays sample document alternatives.  4. The system process the download that click the student.  5. Download finish. |
| Pre-condition | Student download material | |

**Table 23: download material use case description for student**

|  |  |  |
| --- | --- | --- |
| Use case name | Assign coordinator | |
| Use case id | UC 24 | |
| Description | The department head assign coordinator. | |
| Participating actor | Department head | |
| Per condition | The department head must login to the system | |
| Flow of action | Actor action | System response |
| 1. The department head first login to the system.  2. The department head click on the assign coordinators link.  4. The department head select coordinators list from the department.  5. The department head click on assign button.  7. Use case end. | 3. The system display assign coordinators page.  6. The system display assign coordinator successfully. |
| Post condition | The department head assign coordinators successfully | |

**Table 24: assign coordinator use case description for department head**

|  |  |  |
| --- | --- | --- |
| Use case name | Send stages/progress | |
| Use case id | UC25 | |
| Description | This use case describe how the advisor uses this system to send stages/progress for the coordinator | |
| Participating actor | Advisor | |
| Pre-condition | The advisor should have valid user name and password. | |
| Flow of action | Actors action | System action |
| 1. The advisor login to the system.  2. The advisor click on send stages/progress link.  4. The advisor fills all the necessary information on the form.  5. The advisor click on send stages/progress button. | 3. The system displays send stages/progress page.  6. The system displays successful message. |
| Alternative course of action | If the advisor did not fill the form correctly the system displays error message and go back to step 4. | |
| Post condition | The coordinator receive the stages/progress follow up sheet. | |

#### **Table 25: send progress use case description for advisor**

|  |  |  |
| --- | --- | --- |
| Use case name | View stages/progress | |
| Use case id | UC26 | |
| Description | This use case describes how the coordinator uses this system to follow up the activity of advisors and students. | |
| Participating actor | Coordinator | |
| Pre-condition | The coordinator should have valid user name and password. | |
| Flow of action | Actors action | System action |
| 1. The coordinator login to the system.  2. The coordinator click on view progress link.  4. The coordinator view the progress. | 3. The system displays view progress page.  . |
| Post condition | The coordinator follow the advisor and student activity. | |

**Table 26: view progress use case description for coordinator use case**

|  |  |  |
| --- | --- | --- |
| Use case name | View feedback | |
| Use case id | UC27 | |
| Description | This use case describes how the coordinator uses this system to view a complaint for the students. | |
| Participating actor | Student | |
| Pre-condition | The student should have valid user name and password. | |
| Flow of action | Actors action | System action |
| 1. The coordinator clicks on login link.  2. The coordinator click on view feedback link.  4. The coordinator view the feedback.  5. Use case end. | 3. The system displays view feedback page. |
| Post condition | The feedback send successfully. | |

**Table 27: view feedback use case description for coordinator**

|  |  |  |
| --- | --- | --- |
| Use case name | Send approved document | |
| Use case id | UC29 | |
| Description | This use case describes how the advisor to send approved document. | |
| Participating actor | Advisor | |
| Pre-condition | The advisor should have valid user name and password. | |
| Flow of action | Actors action | System action |
| 1. The advisor login to the system.  2. The advisor click on send approved document link.  4. The advisor click on send document button view. | 3. The system the system displays approved document and examiners list.  . |
| Post condition | The follow the advisor and student activity. | |

**Table 28: advisor send approve document**

|  |  |  |
| --- | --- | --- |
| Use case name | Send feedback | |
| Use case id | UC30 | |
| Description | This use case describes how the students send feedback to the coordinator. | |
| Participating actor | Student | |
| Pre-condition | The student should have valid user name and password. | |
| Flow of action | Actors action | System action |
| 1. The student login to the system.  2. The student click on send feedback document link.  4. The student click on send feedback button. | 3. The system displays feedback page  .5The system display feedback send successfully |
| Post condition | Display successfully message | |

**Table 29: student send feedback**

### **2.5.3. Activity diagram**

Activity diagram is another important diagram in UML to describe dynamic aspect of the system. It is basically flow chart to represent the flow from one activity to another activists. The activity can be described as an operation of the system. The purpose of activity diagram is to draw the activity flow of a system, the sequence from one activity to another activity and also describe the parallel, branched and concurrent flow of the system.

**Figure 3: activity diagram for login**



**Figure 4: activity diagram for assign group**

****

**Figure 5: activity diagram for create account**

******Figure 6: activity diagram for activate account**



**Figure 7: activity diagram for add mark**



**Figure 8: activity diagram for generate report**

**Figure 9: activity diagram for submit proposal**

### **2.5.4. Object model**

We draw a class diagram depicting the inheritance relationships and association that exist between the entity objects that we identified earlier. The class diagram focuses mainly on the relationship among application domain concept.

#### **2.5.4.1. Data dictionary**

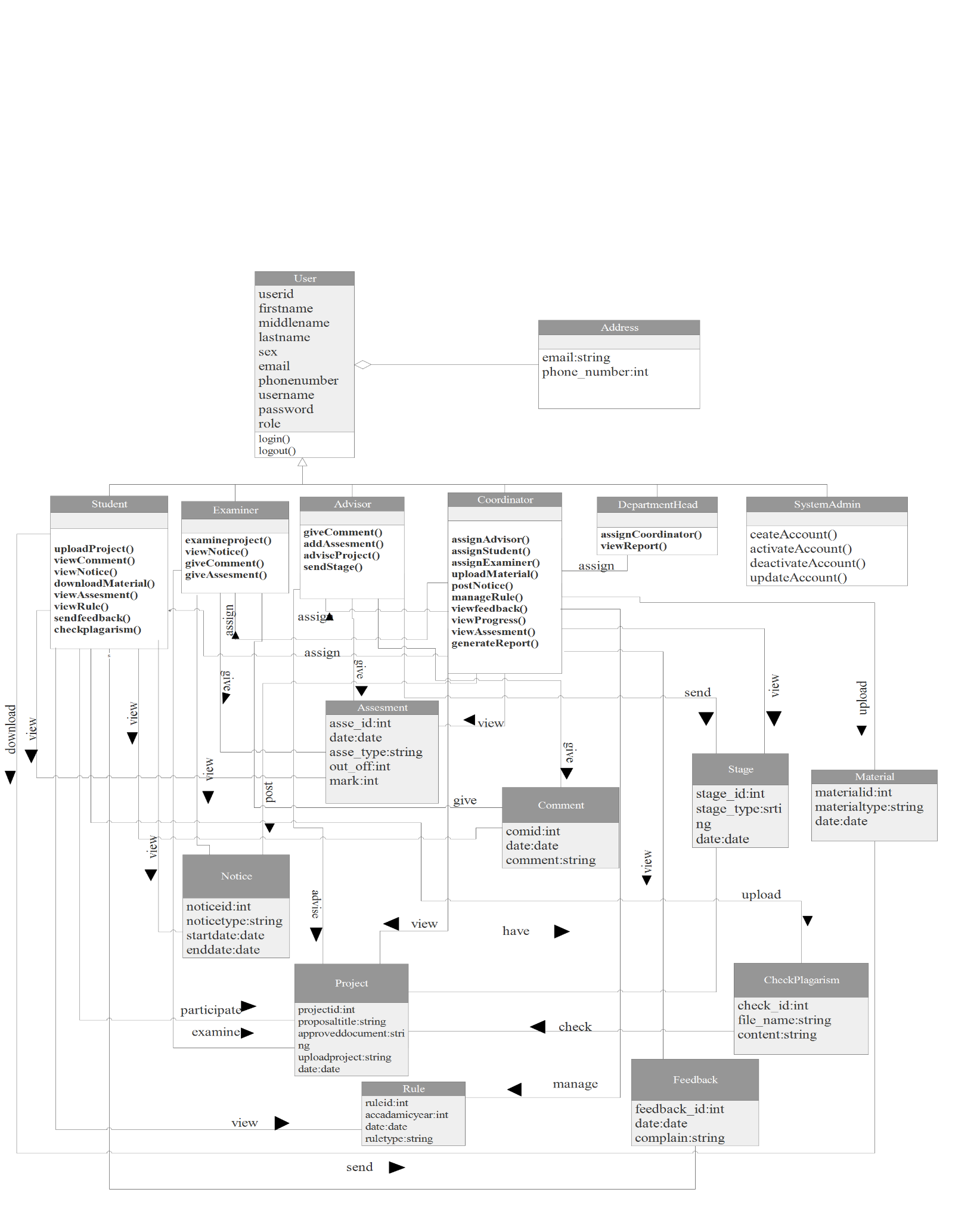
|  |  |  |
| --- | --- | --- |
| Entity object | Attributes | Description |
| System administrator | First name  User name  Password  Email | Users of the system with privileges to administrator create account, deactivate account, update account and activate account. |
| Coordinator | Coordinator id  First name  Last name  Sex  Email | Users of the system with account and privileges to approve title, reject titles, upload material, assign advisor, assign examiner, post schedule. |
| Student | Student id  First name  Last name  Sex  Email | Users of the system with accounts that are privileged to do certain actions in the system like submitting titles, view approved title, view announcement, view posted schedule and download sample document. |
| Advisor | Student id  First name  Last name  Sex  Email | Users of the system with accounts that are assigned to give comment to students on their projects, follow progress, approve completed document and add mark to the student |
| Examiner | Student id  First name  Last name  Sex  Email | Users of the system that are assigned to view posted schedule and add marks. |
| Department head | Student id  First name  Last name  Sex  Email | User of the system with account and privilege to assign coordinator and view generated report. |
| Check plagiarism | Check id | Is used to check project proposal if there is any similarities between project proposals. |
| Notice | notice id  Start date  End date  Time | Itis a plan of student presentation date, time and room. This plan will be post by coordinator. |

**Table 30: data dictionary**

#### **2.5.4.2. Class modeling**

In software engineering, a class diagram in the UML is a type of static structure diagram that describes the structure of a system’s classes, their attributes, operation (methods), and the relationship among objects.

The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling translating the model into programming code.



**Figure 10: analysis class model**

### **2.5.5. Dynamic modeling**

#### **2.5.5.1. State chart diagram**

****State chart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different state of an object during its life time and this states are changed by events. State chart diagram describes the flow of control from one sate to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of state chart diagram is to model lifetime of an object from creation to termination [4].

Figure 11: state chart diagram for checkplagarism

#### **2.5.5.2. Sequential diagram**

A sequential diagram shows object interactions arranged in the time sequence. It depict the objects and the classes involved in the scenario and the sequence of message exchanged between the object needed to carry out the functionality of the scenario. Sequential diagrams are typically associated with use case realizations in the logical view of the system under development. Sequential diagrams are sometimes called event diagram or event scenario.



**Figure 12: sequential diagram for assign coordinator**



**Figure 13: sequential diagram for assign group**



**Figure 14: sequential diagram for submit proposal**



**Figure 15: sequential diagram for checking plagiarism**



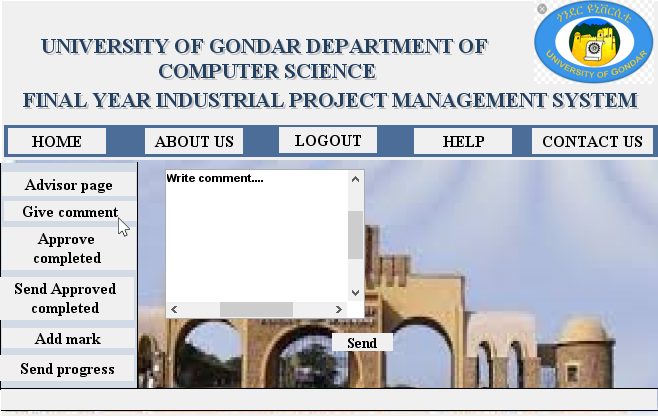
**Figure 16: sequential diagram for approve completed document**

#### **2.5.5.3. User Interface**

User interface (UI) is everything designed into an information device with which a person may interact. It is also the way through which user interacts with an application or a website. The goal of User Interface design is to produce a user interface which makes it easy, efficient, and enjoy able (user-friendly) [5].



**Figure 17: user interface for login**



**Figure 18: user interface for advisor give comment**

# **Chapter 3**

# **3. System design**

## **3.1. Introduction**

Compared with requirements analysis system design is messy. One of the reason is that the requirement analysis depends on the application (problem) domain but system design is the first step to get in to the solution domain. System design is concerned about the overall structure of the system. The beast system design is one where the interaction between is minimal.

Mapping the subsystem to processors is the moment or truth, and the hardest. While decomposition and concurrency identification are still independent of technology, subsystem allocation is not. Shall we map them to hardware or can we get by with software. The management of data also is the design issue.

Access methods specify the security of the design. Can a random user or program gone hay-wire create havoc to the rest of the system. Control is another important design issue. The purpose of the system design is to supplement the system architecture providing information and data useful and necessary for implementation of the system elements. Design definition is the process of developing, expressing, documenting, and communicating the realization of the architecture of the system through a complete state of design characteristics describing inform suitable for implementation [6].

The design goals are derived from the non-functional requirements of the system. It describes the qualities of the system. Generally, the objectives of design are to model the system with high quality. The purpose of design is to find out how we are going to build the system and to obtain the information needed to get the actual implementation of our system. Some design goals are:

**Performance:** this shows how the system is expected to behave under normal operating condition. Our system can perform essential activity found in the FYIPMS with higher performance that the system responds to operation in seconds upon request in order to make the system effective with regard to time, improving the process speed of the system.

**Usability:** the user of the system will achieve their operation successfully and the system will give satisfaction to these who use the system. Our system will be easy to use and easy to learn.

**Robustness**: the software handle in valid user input. This can be done by using different type of exception handling mechanism. This enable the software not to be suspended even if the user invalid input.

**Reliability:** the code will be easily readable and understandable by other programmers. To achieve this goal we will supplement the code by proper naming, connection, proper indentation and comment.

**Security:** the system should be secured. This means that our system does not allow unauthorized users to access the data that has no the right to access it.

## **3.2. Current software architecture**

Since the current system is manual based system there is no current software architecture.

## **3.3. Proposed software architecture**

### **3.3.1. Overview**

In our project we use three tire architecture. A three tire architecture is client-server architecture in which the functional process logic, data access, computer data storage and user interface are developed and maintained as independent modules on separate platforms. The three tires in three tire architecture are: Presentation tire occupies the top level and displays information related to services available on a website. This tier communicates with other tiers by sending results to the browser and other tiers in the network. The application tire also called the middle tier, logical tier, business logic, this tier is pulled from the presentation tier. It controls application functionality by performing detail processing. The third tier is the data tier which is house database servers where information is stored and retrieved. Data in this tier is kept independent of application servers [7].



**Figure 19: three tire architecture**

### **3.3.2. Subsystem decomposition**

System may be built from component in component based architecture. To describe subsystem decomposition we use component diagram. Component diagram shows how object (classes) in our system grouped together and form component.

The component interact with each other either in giving service to other component or requesting service from other component. The component diagram helps to model the physical aspect of an object-oriented software system. It illustrate the architectures of the software component and the dependencies between them. Those software components including run- time, executable also the source code component. The component diagrams main purpose is to show the structure relationships between the components of a system [8].



**Figure 20: sub system decomposition of FYIPMS by using component diagram**

### **3.3.3. Hardware and software mapping**

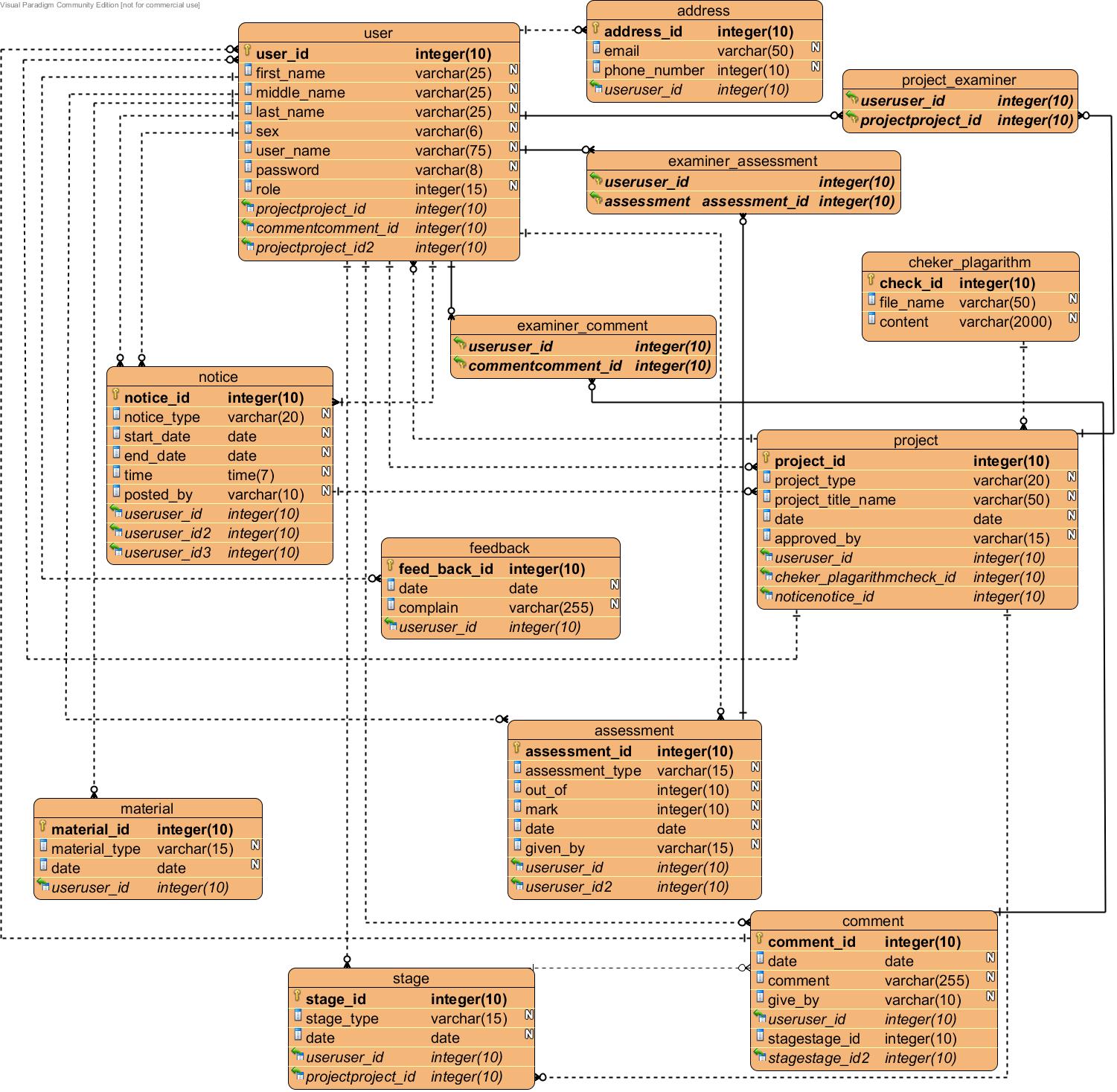
**A deployment diagram** in the Unified Modeling Language models the physical deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database). The web based part is run on a networked environment on different Operating System platforms. The client/server architecture of the system enables different clients to connect to the server remotely through Intranet connection. Server side there is web server that is always connected with the intranet for listening HTTP requests and accepts connection request and uses Apache HTTP server manipulates data from the database using PHP programs and answers user’s request [9].



**Figure 21: deployment diagram for FYIPMS**

### **3.3.4. Persistent data management**

Persistent data management describes the persistent data stored by the system and the data management infrastructure required for it.

****

**Figure 22: persistent data management for FYIPMS**

### **3.3.5. Access control and security**

Access control is a security technique that can be used to regulate who or what can view or use resources in a computing environment. In this system, different actors have access to different functionalities and data. Therefore these privileges put off unauthorized users from accessing data’s which they don’t have privacy to access.

**Authentication:** The process of ascertaining that somebody really is who he claims to be. In these system users before entering into the system they must be authenticated as authorized users. This takes place by letting users to insert their username and password in the displayed login form.

**Authorization:** Permission to access resource, riles that determine who is allowed to do what. After authentication users are granted for specific tasks. This take place by preventing users from participating in specific tasks on which he/she doesn’t have grant to access.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Actor**  **Class** | **Department head** | **Coordinator** | **Student** | **Advisor** | **Examiner** | **System admin** |
| **Group** |  | Assign group() |  |  |  |  |
| **Project** |  |  | Send project()  Submit proposal() | Approve completed project() |  |  |
| **Notice** |  | Post notice() | View notice |  | View notice() |  |
| **Progress** |  | View progress() |  | Send progress() |  |  |
| **Rule** |  | Manage rule() | View rule() |  |  |  |
| **Material** |  | Upload material() | Download material() |  |  |  |
| **Mark** |  | View assessment() | View assessment() | Add assessment() | Add assessment() |  |
| **Comment** |  |  | View comment() | Give comment() | Give comment() |  |
| **Check plagiarism** |  |  | Upload proposal() |  |  |  |
| **Report** |  | Generate report() |  |  |  |  |
| **Account** |  |  |  |  |  | Create,Activate, deactivate,Update account() |

**Table 31: access control**

### **3.3.6 Subsystem service**

This section describes the service provided by each sub system.

**Project subsystem**: this subsystem is used to coordinator, advisor, and examiner assign group, assign examiner, assign advisor, post notice, view progress, view notice, report generate, add assessment, approve completed project, give comment, send progress, manage rule, upload material, view assessment.

**Student subsystem**: this subsystem is used to submit project proposal with group, send completed project, and view assessment.

**Plagiarism subsystem**: this subsystem is used to check if there is any similarity between the proposed project and the previous projects that are done before.

**Department subsystem**: this subsystem is used to assign coordinator, view generated report.

**User subsystem**: this subsystem is manages users, it contains users information as well as some action that perform by users such as Login operation to check the user is valid or not control unauthorized users before entrance to the system. It consists of the authorization, autontication and security of the system.

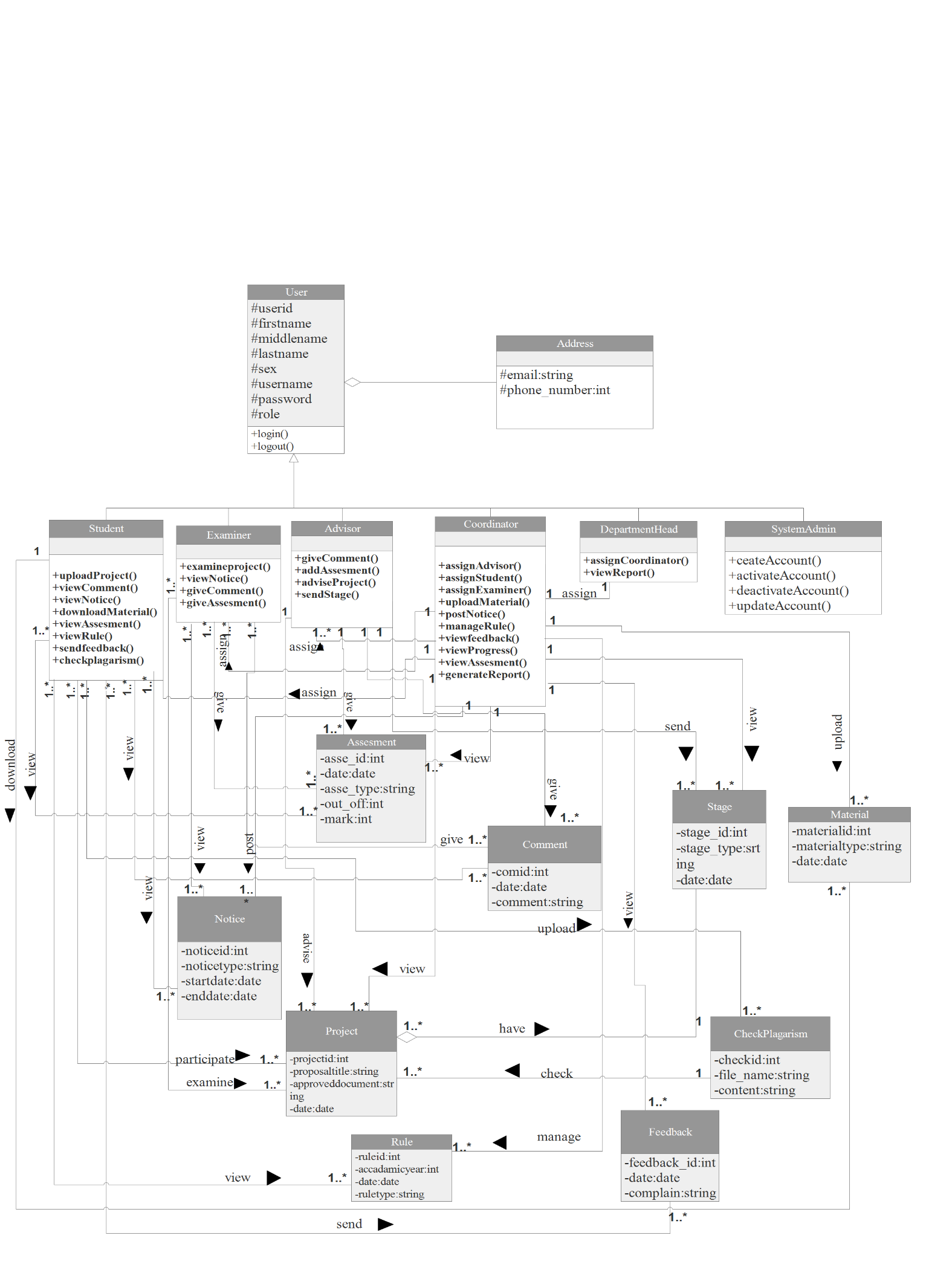
Logout operation is recommended to properly logout after finishing activities in the system.

**Account subsystem**: this subsystem is used to the system admin to create, activate, deactivate and update users account.

## **3.4. Detail Design**

### **3.4.1. Design class model**

The below diagram shows the detail class diagram which shows the relationship between classes their attribute, operation and visibilities and also there relation includes multiplicity and inheritance or associations are shown. Let us see their class and relation in this system:



**Figure 23: detail class diagram**

### **3.4.2. Package**

This section describes the decomposition of subsystems into packages and file organization of the code. This includes an overview of each package, its dependencies with other packages, and its expected usage.



**Figure 24: package diagram**

# **CONCLUSION**

Final Year Industrial Project Management System is one of the main work in UOG department of computer science. This system is a web based application to serve the user in department of computer science. This system is to simplify the work load of coordinator and save time to student. The system developed in the project consists of web based applications. The web based application facilitates, check if there is any similarity between project proposals student submit proposal with group, coordinator assign advisor and examiner, post schedule, Enable students and advisors to contact each group in face to face which used to decrease the gap between student and advisor. Upload previous documents/projects and materials/guidelines online. Examiner and advisor give students assessment online and student can view mark. Through various challenging; now the team is coming to the end of this project. Those different challenges made possible by the cooperation of all the group members. Developing this project all group members contributed their full capability with maximum interest and all group members get ways toward developing a project.

# **REFERENCE**

[1]. http://www.UOG.edu.et

[2]. http://search software quality techtarget.com/definition/iterative-development

[3].whatis.techtarget.com/definition/use-case-diagram

[4].http://searchmicroservice.techtarget.com/definition//state-diagram-state-chart-diagram

[5].http://searchingmicroservice.techtarget.com/definition

[6].objective-oriented-software-engineering using UML-patterns, and java system design.

[7].http://www.techopedia.com/24649/three-tire-architecture

[8].http://www.visual-paradigm.com/guide/uml-unified-modeling-language/whatis component-diagram

[9].http://en.wikipedia.org/wiki/deployment diagram

# **APPENDEX A**

