AMS

0.1v

ALUMNI MANAGEMENT SYSTEM

The purpose of this document is to provide with a template for documenting AMS.

**Document Control :**

| **Project Revision History** | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | |  |  |  |  |  |
| **Date** | **Version** | **Author** | **Brief Description of Changes** | | | | **Approver Signature** | |
| 11-11-2022 | 0.1v | Group-02 | Initial Draft | | | |  | |
| 14-11-2022 | 0.2v | Group-02 | Added DFD diagrams | | | |  | |

[**1. Introduction 5**](#_heading=h.3znysh7)

[1.2. Acronyms/Abbreviations 5](#_heading=h.tyjcwt)

[1.1. Intended Audience 5](#_heading=h.2et92p0)

[1.3. Project Purpose 5](#_heading=h.3dy6vkm)

[1.4. Key Project Objectives 5](#_heading=h.1t3h5sf)

[1.5. Project Scope and Limitation](#_heading=h.4d34og8) 6

1.5.1. In Scope 6

1.5.2. Out Of Scope [*6*](#_heading=h.2s8eyo1)

[1.6. Functional Overview 6](#_heading=h.17dp8vu)

[1.7. Assumptions, Dependencies & Constraints 6](#_heading=h.3rdcrjn)

[**2. Design Overview 6**](#_heading=h.26in1rg)

[2.1. Design Objectives 6](#_heading=h.lnxbz9)

[*2.1.1. Recommended Architecture 7*](#_heading=h.35nkun2)

[2.2. Architectural Strategies 7](#_heading=h.1ksv4uv)

[*2.2.1. Design Alternative 8*](#_heading=h.44sinio)

[*2.2.2. Reuse of Existing Common Services/Utilities 8*](#_heading=h.2jxsxqh)

[*2.2.3. Creation of New Common Services/Utilities 9*](#_heading=h.z337ya)

[*2.2.4. User Interface Paradigms 9*](#_heading=h.3j2qqm3)

[*2.2.5. System Interface Paradigms 9*](#_heading=h.1y810tw)

[*2.2.6. Error Detection / Exceptional Handling 9*](#_heading=h.4i7ojhp)

[*2.2.7. Memory Management 9*](#_heading=h.2xcytpi)

[*2.2.8. Performance 10*](#_heading=h.1ci93xb)

[*2.2.9. Security 10*](#_heading=h.3whwml4)

[*2.2.10. Concurrency and Synchronization 10*](#_heading=h.2bn6wsx)

[*2.2.11. Housekeeping and Maintenance 10*](#_heading=h.qsh70q)

[**3. System Architecture 11**](#_heading=h.3as4poj)

[3.1 System Architecture Diagram. 11](#_heading=h.1pxezwc)

[3.2System Use-Case 11](#_heading=h.49x2ik5)

[3.1. Subsystem Architecture 13](#_heading=h.2p2csry)

[3.2. System Interfaces 13](#_heading=h.147n2zr)

[*3.2.1. Internal Interfaces 13*](#_heading=h.3o7alnk)

[*3.2.2. External Interfaces 13*](#_heading=h.23ckvvd)

[**4. Detailed System Design 14**](#_heading=h.ihv636)

[**Already Mentioned 14**](#_heading=h.2iq8gzs)

[4.1. Key Entities 14](#_heading=h.32hioqz)

[4.2. Detailed-Level Database Design 14](#_heading=h.1hmsyys)

[*4.2.1. Data Mapping Information 17*](#_heading=h.41mghml)

[*4.2.2. Data Conversion 17*](#_heading=h.2grqrue)

[4.3. Archival and retention requirements 17](#_heading=h.vx1227)

[4.4. Disaster and Failure Recovery 17](#_heading=h.3fwokq0)

[4.5. Business Process workflow 17](#_heading=h.1v1yuxt)

[4.6. Business Process Modeling and Management (as applicable) 17](#_heading=h.4f1mdlm)

[4.7. Business Logic 17](#_heading=h.2u6wntf)

[4.8. Variables 18](#_heading=h.19c6y18)

[4.9. Activity / Class Diagrams (as applicable) 18](#_heading=h.3tbugp1)

[4.10. Data Migration 18](#_heading=h.28h4qwu)

[*4.10.1. Architectural Representation 18*](#_heading=h.nmf14n)

[*4.10.2. Logical View 19*](#_heading=h.1mrcu09)

[*4.10.3. Architecturally Significant Design Packages 19*](#_heading=h.46r0co2)

[*4.10.4. Data model 19*](#_heading=h.2lwamvv)

[*4.10.5. Deployment View 19*](#_heading=h.111kx3o)

[**5. Environment Description 20**](#_heading=h.3l18frh)

[5.1. Time Zone Support 20](#_heading=h.206ipza)

[5.2. Language Support 20](#_heading=h.4k668n3)

[5.3. User Desktop Requirements 20](#_heading=h.2zbgiuw)

[5.4. Server-Side Requirements 20](#_heading=h.1egqt2p)

[*5.4.1. Deployment Considerations 20*](#_heading=h.3ygebqi)

[*5.4.2. Application Server Disk Space 20*](#_heading=h.2dlolyb)

[*5.4.3. Database Server Disk Space 20*](#_heading=h.sqyw64)

[*5.4.4. Integration Requirements 20*](#_heading=h.3cqmetx)

[*5.4.5. Jobs 20*](#_heading=h.1rvwp1q)

[*5.4.6. Network 20*](#_heading=h.4bvk7pj)

[*5.4.7. Others 20*](#_heading=h.2r0uhxc)

[5.5. Configuration 20](#_heading=h.1664s55)

[*5.5.1. Operating System 20*](#_heading=h.3q5sasy)

[*5.5.2. Database 21*](#_heading=h.25b2l0r)

[*5.5.3. Network 21*](#_heading=h.kgcv8k)

[*5.5.4. Desktop 21*](#_heading=h.34g0dwd)

[**6. References 21**](#_heading=h.1jlao46)

[**7. Appendix 21**](#_heading=h.43ky6rz)

# 

# 1. INTRODUCTION

The Alumni Management System is a simple C Project in which a university can keep track with its students and graduates. Students can view their profile and alumni can create a job information. The main aim of this project is to make connections between alumni and students. The project manages the fresh as well as old graduate students with their respective information in actively participating in registering, searching and managing the alumni information for job opportunities and resources.

## 1.1.Intended Audience

This document is viewed across the following members CG Employees, BU SME’s and internal SME’s

| CG Employees |  |
| --- | --- |
| BU SME |  |

## 1.2.Acronyms/Abbreviations

| AMS | Alumni Management System |
| --- | --- |
| DFD | Data Flow Diagram |

## 1.3.Project Purpose

The purpose of the project is to connect the students with the alumni for any job post and the alumni can post the job updates or any information relate to job and can share the materials relate to the job. Students can connect to particular alumni for job. This is more beneficial for students who are in search of job for their career.

## 1.4. Key Project Objectives

In this project student can interact with the alumni directly if there are any job updates posted by the alumni.

Student can register by providing some details and login by using the same details to access the alumni details.

In the same way alumni can also register and login to give more updates for the students regarding the job updates or any materials related to the job.

## 1.5. Project Scope and Limitation

**1.5.1. In Scope**

This project is related to console application not network application.

### 1.5.2. Out of scope

Not Applicable

## 1.6. Functional Overview

There are three different types of modules present in this project they are alumni, student and admin. There are different functions included for every user for example if alumni are using this portal, then he can do different type of actions like alumni can register in this portal and can login by using the information and he/she can post job and materials related to jobs which can be accessed by the students which are useful for building their career. In the same way Student can also register and login in this portal and connect with multiple alumni who are registered in the portal. Admin can edit the alumni and delete the alumni and can view all the connections between the student and the alumni.

## 1.7. Assumptions, Dependencies & Constraints

* The system should contain Linux installed.
* This project work on laptop/desktop.

**1.8. Risks**

No risks are identified in this project

# 2. DESIGN OVERVIEW

University Alumni Management System is a Software and it managing the alumni and student data by keeping all the data in just one software. In this software Alumni and students can view and edit their profile. Admin can edit and delete the alumni details. Also, we are providing an option for alumni to post a job recruitment and this can also be viewed by students as well as the alumni who have experienced and wanted a job. Students can also see the current job offers, a list of which will be displayed. Students can ask for a connect with alumni using ask for connect function. Students can access all the information posted by alumni like job updates/technical advice/Study materials.

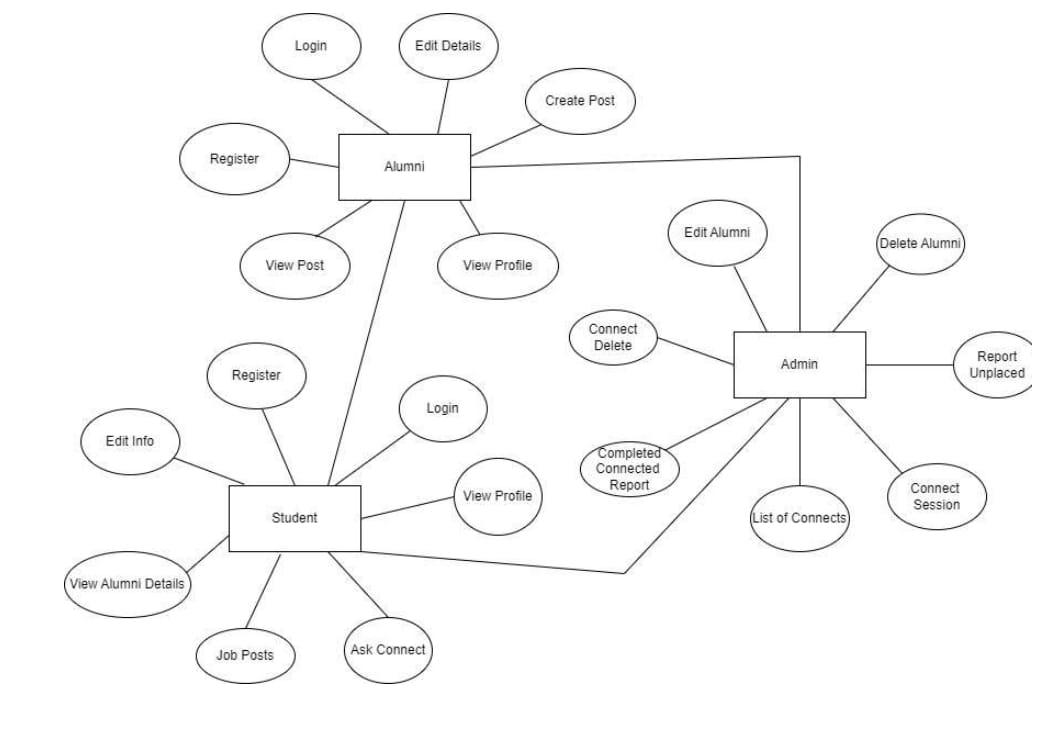
## 2.1. Design Objectives

There are three major modules:

* Student
* Alumni
* Admin

### 2.1.1. Recommended Architecture

The Recommend Architecture of Alumni Management System is:



## 2.2. Architectural Strategies

**ALUMNI:**

* Alumni Registration
* Login
* Update Profile
* View Details
* Create Posts
* View Posts

**STUDENT:**

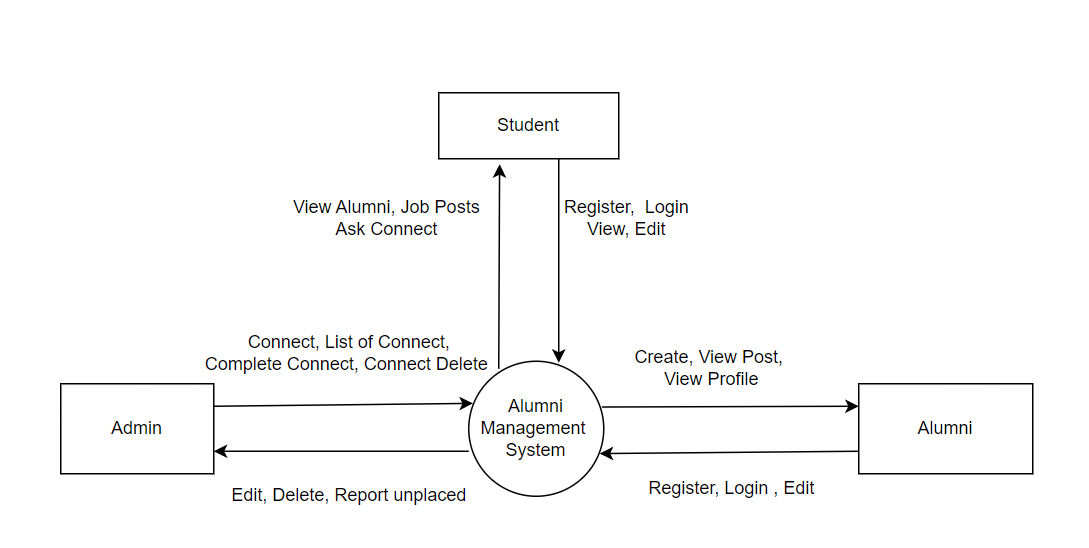
* Student Login
* Add/Edit Details
* View job Details
* View Alumni Details
* View profile
* Ask for connect

**ADMIN:**

* Edit\_Alumni
* Delete \_Alumni\_Details
* Report\_placed\_Students
* Connect\_session
* List of Connects
* Completed\_connect\_Report
* Connect\_Delete

### 2.2.1. Design Alternative

The Alternative Design for our Project is:



### 2.2.2. Reuse of Existing Common Services/Utilities

Given an existing component, one identifies all the services that it provides or requires, analyzes each service in the target system architecture, considers service allocation to subsystems or components, selects possible integration techniques, and determines services to be reused. In the case where the system architecture requires a system component to provide the same service that an existing component already provides, one performs trade analysis between reuse of the existing service and developing the service in a new subsystem or component according to overall project constraints and benefits. Throughout the paper, the author uses an example of developing a system.

### 2.2.3. Creation of New Common Services/Utilities

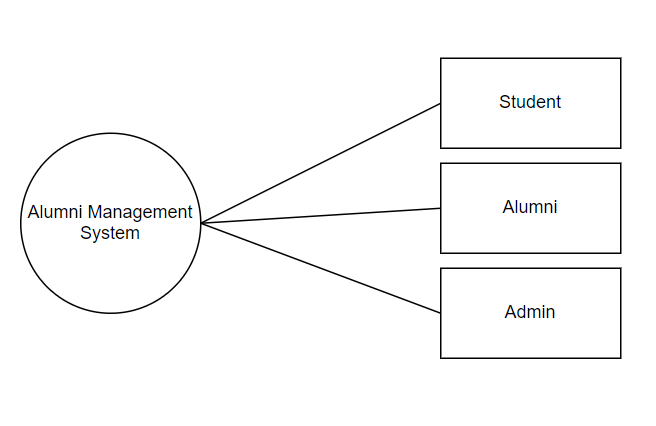
Not applicable – Since we have not created any common services and Utilities.

### 2.2.4. User Interface Paradigms

User Interface Paradigms A user interface is how a program (including the operating system) communicates with a person. The two common interfaces are the Console and the Graphical User Interface (abbreviated GUI and pronounced gooey). Each of these has a different approach that needs to be considered.

### 2.2.5. System Interface Paradigms

The Architecture of System Interface Paradigm is:



### 2.2.6. Error Detection / Exceptional Handling

Not Applicable.

### 2.2.7. Memory Management

In this University Alumni Management system data are stored in System safe and in a secure manner. Memory is always used to store the data collected from the alumni and the student. Once the data is deleted by the admin, it will be deleted from memory also. We will use file management for storing the data of students and alumni.

To achieve a degree of multiprogramming and proper utilization of memory, memory management is important. Many memory management methods exist, reflecting various approaches, and the effectiveness of each algorithm depends on the situation.

### 2.2.8. Performance

* The Alumni Management System is reliable.
* This project will be very responsive and fast.
* This will not take much space of the system.
* This project is optimized in every aspect thus works finely.
* This platform enables organizations to handle passwords secure to login and other users can’t access personal details of other alumni or students.

### 2.2.9. Security

This system designed with good security principles ensures Integrity of the system and prevent security problems.

### 2.2.10. Concurrency and Synchronization

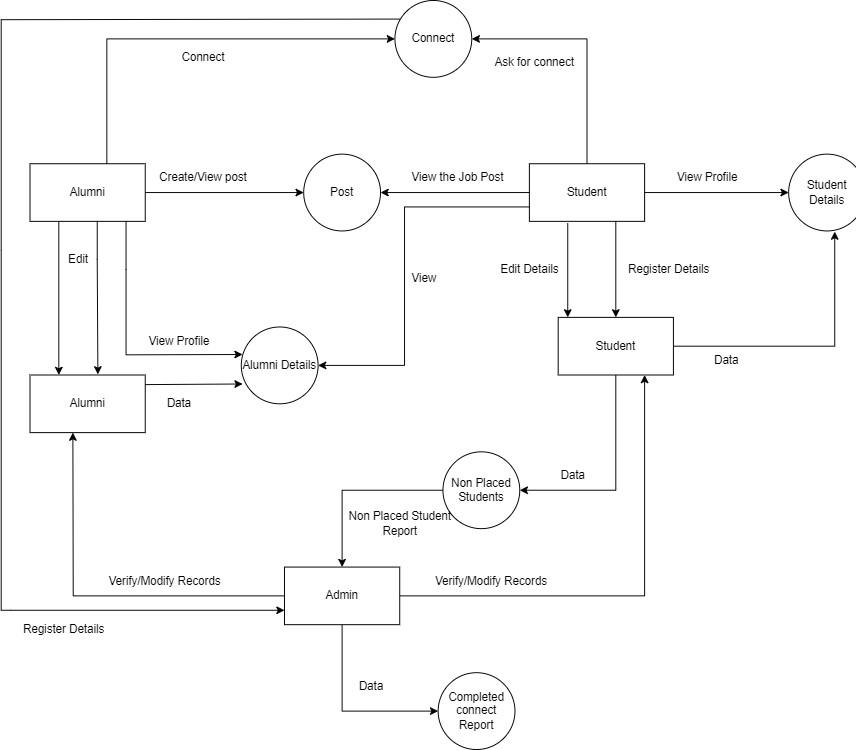
Not Applicable

### 2.2.11. Housekeeping and Maintenance

Not Applicable

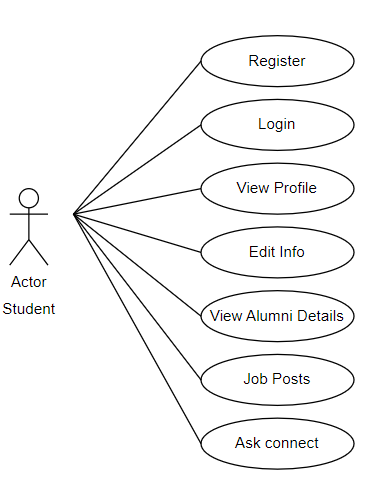
# 3. System Architecture

## 3.1 System Architecture Diagram.

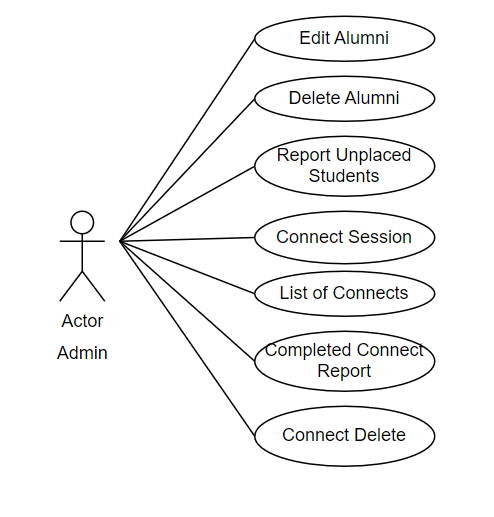


## 3.2 System Use-Case

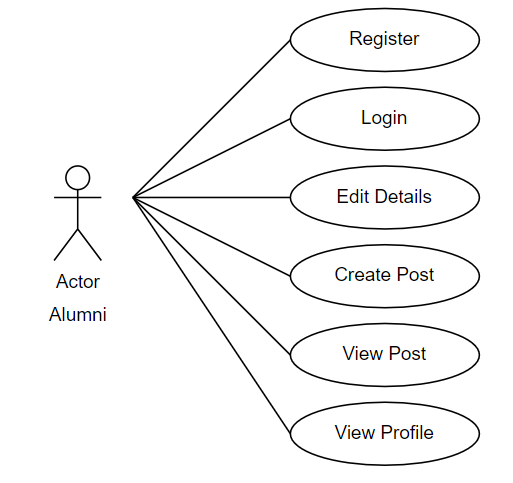
**Student Use-Case:**



**Admin Use-Case:**



**Alumni Use-Case:**



## 3.3. Subsystem Architecture

Not Applicable

## 3.4. System Interfaces

Already mentioned

### 3.4.1. Internal Interfaces

Not Applicable

### 3.4.2. External Interfaces

Not Applicable

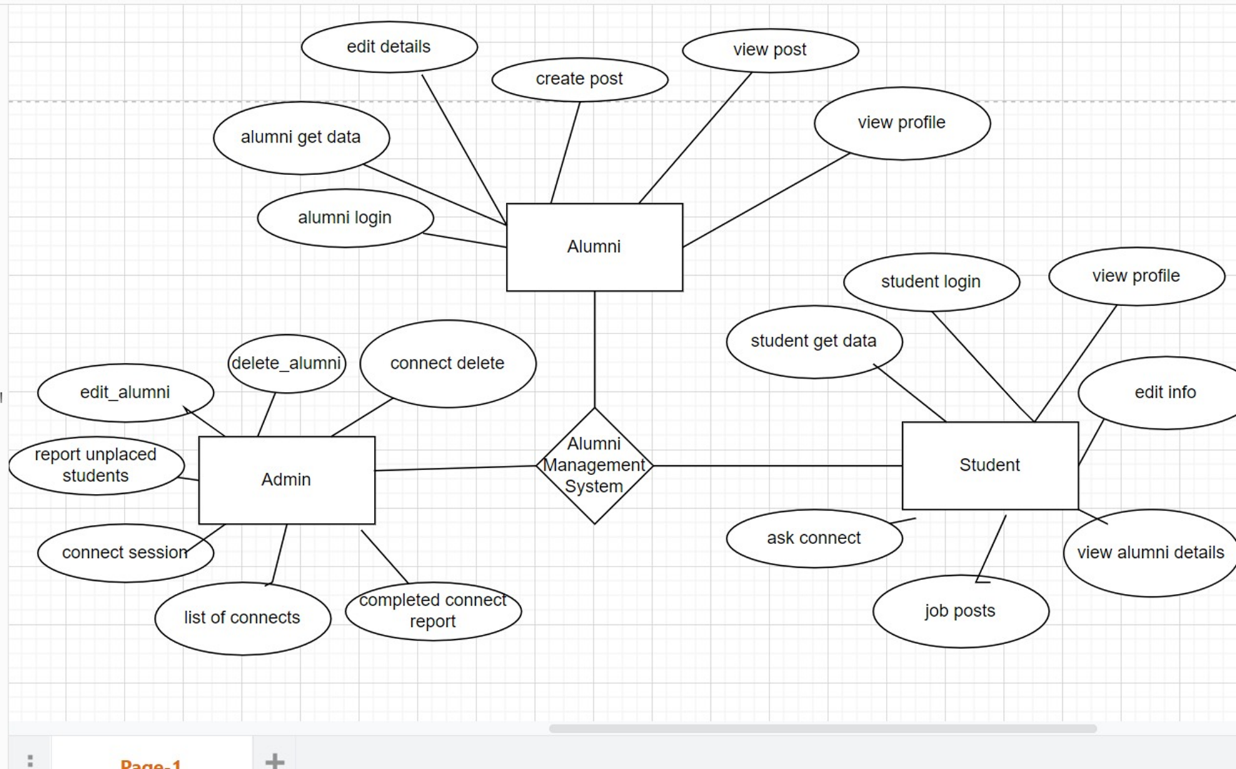
# 4. DETAILED SYSTEM DESIGN

Already Mentioned

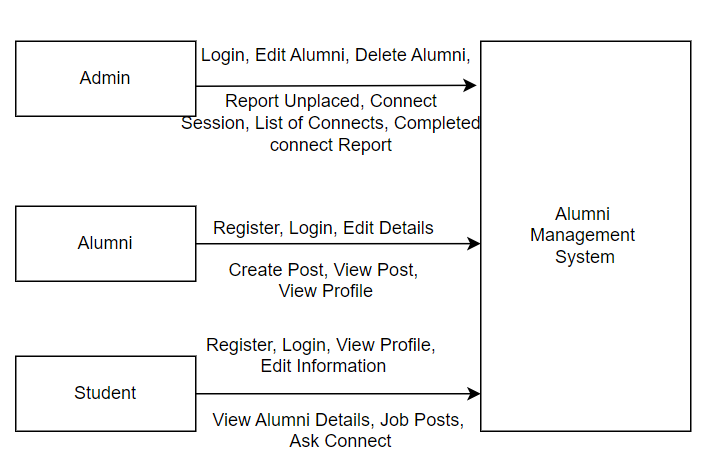
## 4.1. Key Entities

Already Mentioned

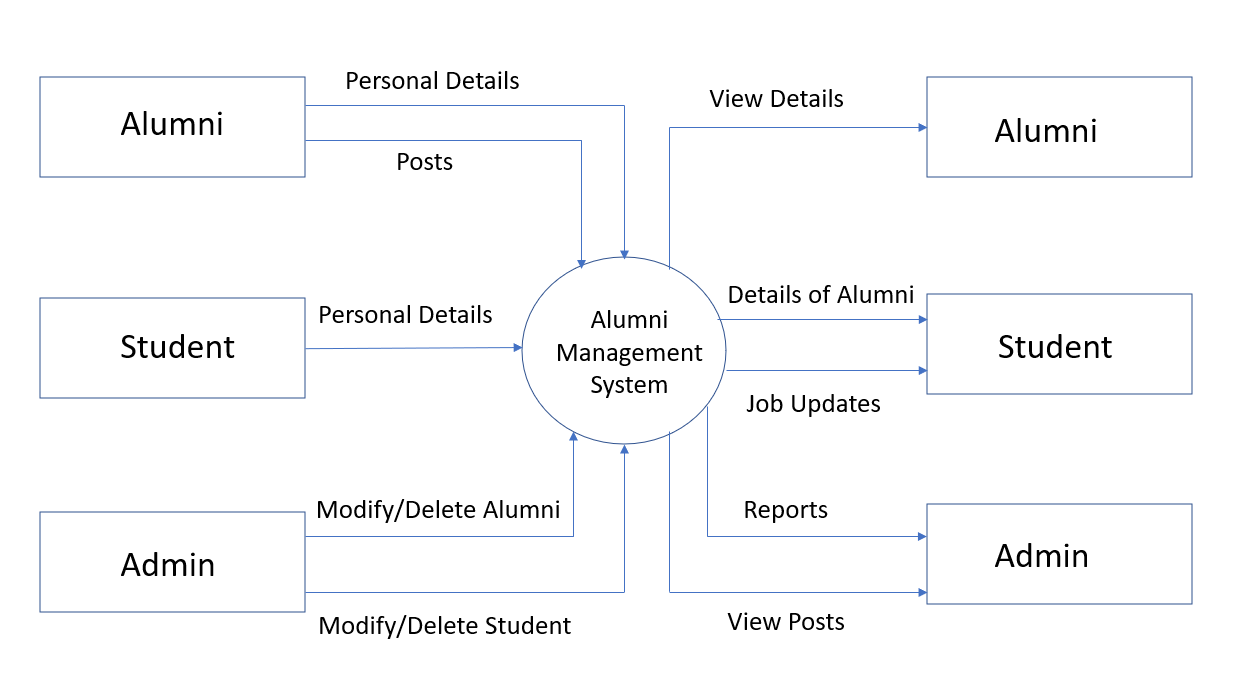
## 4.2. Detailed-Level Database Design

**

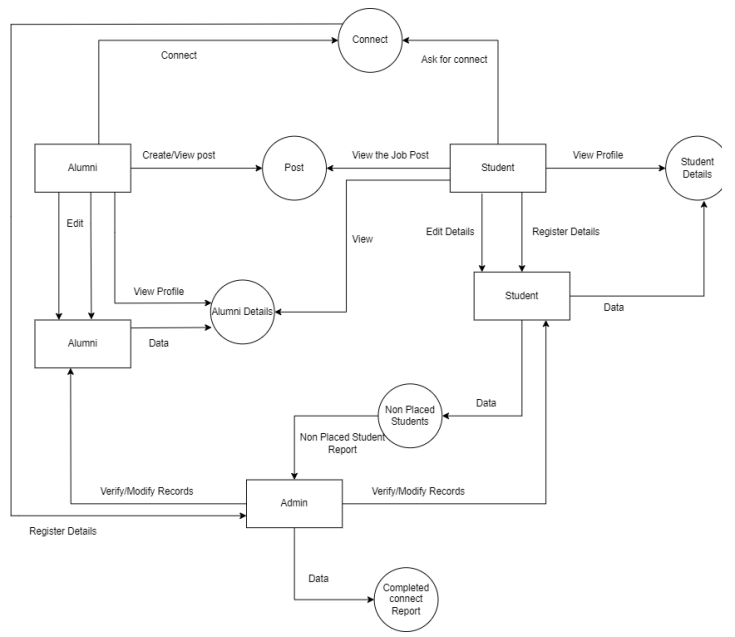
**DFD Level 0:**

****

**DFD Level 1:**



**DFD Level 2:**



### 4.2.1. Data Mapping Information

Not Applicable

### 4.2.2. Data Conversion

Not Applicable

## 4.3. Archival and retention requirements

Not Applicable

## 4.4. Disaster and Failure Recovery

Not Applicable

## 4.5. Business Process workflow

Not Applicable

## 4.6. Business Process Modeling and Management (as applicable)

Not Applicable

## 4.7.Business Logic

Not Applicable

## 4.8. Variables

All the naming convention of variables and functions are easily understandable

For the new project developer thus naming is universally done.

## 4.9. Activity / Class Diagrams (as applicable)

Not Applicable

## 4.10. Data Migration

Not Applicable

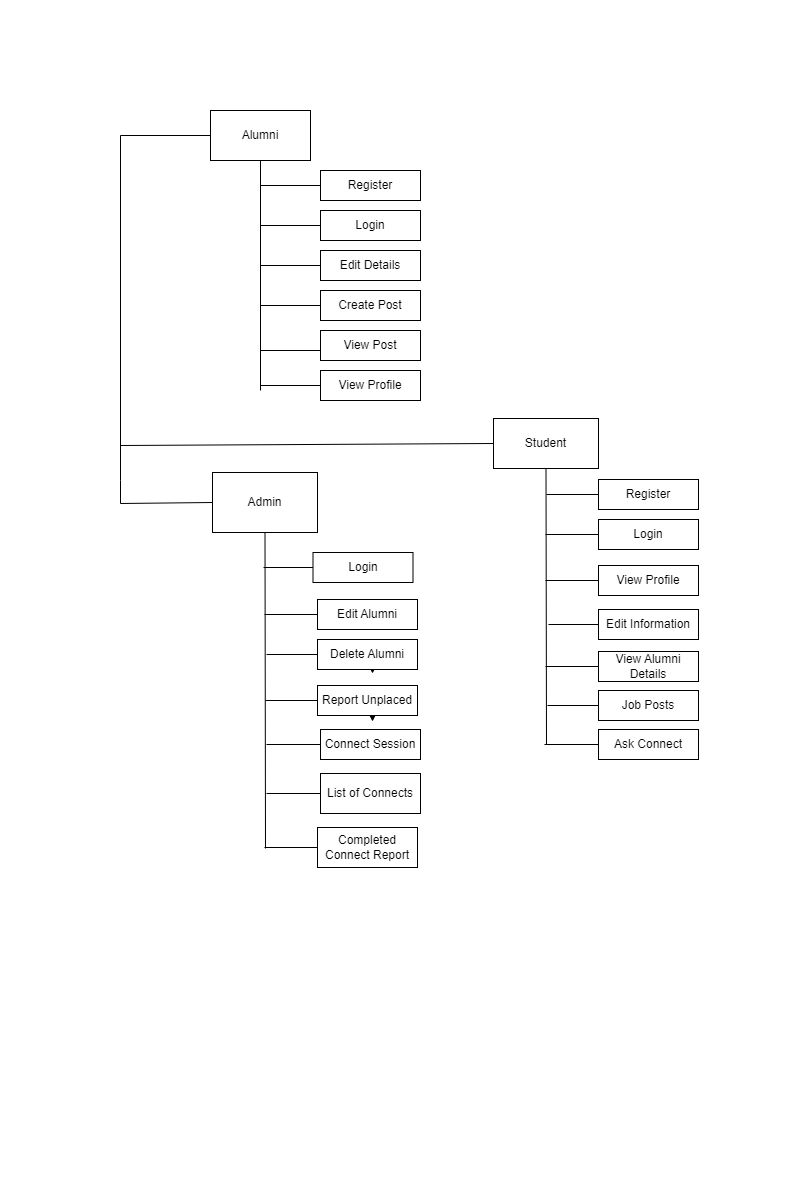
### 4.10.1. Architectural Representation

Already Mentioned

**4.10.2. Architectural Goals and Constraints**

Not Applicable

### 4.10.3. Logical View

**

### 4.10.4. Architecturally Significant Design Packages

Not Applicable

### 4.10.5. Data model

Not Applicable

### 4.10.6. Deployment View

Not Applicable

# 5. ENVIRONMENT DESCRIPTION

## 5.1. Time Zone Support

It supports the time zone as per Indian Standard Time (IST).

## 5.2. Language Support

This project only supports the English language.

## 5.3. User Desktop Requirements

The requirement for users’ desktop is Linux should be installed.

## 5.4. Server-Side Requirements

Not Applicable

### 5.4.1. Deployment Considerations

Not Applicable

### 5.4.2. Application Server Disk Space

Not Applicable

### 5.4.3. Database Server Disk Space

Not Applicable

### 5.4.4. Integration Requirements

Not Applicable

### 5.4.5. Jobs

Not Applicable

### 5.4.6.Network

Not Applicable

### 5.4..7.Others

Not Applicable

## 5.5. Configuration

### 5.5.1. Operating System

Linux to be installed.

4 gb RAM.

I3/i5 processor.

### 5.5.2. Database

Not Applicable

### 5.5.3. Network

Not Applicable

### 5.5.4. Desktop

Windows 10

# 6. References

<https://www.irjet.net/archives/V9/i4/IRJET-V9I4445.pdf>

<https://www.sourcecodester.com/php/14524/alumni-management-system-using-phpsql/>

# 7. Appendix

<https://www.geeksforgeeks.org/basics-file-handling-c/>

<https://www.intelschool.in/alumni-management-system/>

**Change Log**

| **QMS Template Version Control (Maintained by QA)** | | | | | |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| **Date** | **Version** | **Author** | | **Description** | |
| 11-11-2022 | 0.1V | Group-02 | | Initial Draft | |
| 14-11-2022 | 0.2v | Group-02 | | Added DFD diagrams | |