# **Data Design:**

Data design is a major part of the proposed solution. This section describes the study of the basic information such as the data description covering all the aspects of entities etc. The data dictionary is also covered in this section and the database description is also defined. As the application is helping in the providing the proper guide of the university timeline therefore the information is aligned according to such scenario.

# **Data Description:**

The data description in this case provides the information of the entities, fields, and the datatypes for the case of the app design concerning the university guide. The complete information can be seen below:

## **Student:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default |
| User\_ID | decimal(50,0) | No | PRI | NULL |
| User\_type | varchar(45) | No |  | NULL |
| Password | decimal(50,0) | No |  | NULL |
| Education | varchar(45) | No |  | NULL |
| Day | date | No |  | NULL |
| Month | date | No |  | NULL |
| Year | date | No |  | NULL |
| Address | varchar(45) | No | MUL | NULL |
| Course | varchar(45) | No | MUL | NULL |
| Email | varchar(45) | No |  | NULL |
| Facility | varchar(45) | No | MUL | NULL |
| Phone | varchar(15) | No | PRI | NULL |

## **Alumnus:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default |
| User\_ID | decimal(50,0) | No | PRI | Null |
| User\_name | varchar(35) | No |  | Null |
| User\_Pass | int | No |  | Null |
| User\_Role | varchar(35) | No |  | Null |
| Level | varchar(35) | No |  | Null |
| Description | varchar(35) | No |  | Null |
| Education\_level | varchar(35) | No |  | Null |
| GPA | float | No |  | Null |
| Roll\_num | float | No |  | Null |
| Pic | varchar(35) | No |  | Null |
| College\_ID | decimal(50,0) | No | MUL | Null |

## **Visitor:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default |
| User\_type | varchar(45) | No | PRI | NULL |
| User\_ID | decimal(50,0) | No | PRI | NULL |

## **Faculty:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default |
| User\_ID | decimal(50,0) | No | PRI | Null |
| User\_name | varchar(35) | No |  | Null |
| User\_Pass | int | No |  | Null |
| User\_Role | varchar(35) | No |  | Null |
| Level | varchar(35) | No |  | Null |
| Description | varchar(35) | No |  | Null |
| Pic | varchar(35) | No |  | Null |
| Faculty\_ID | decimal(50,0) | No | MUL | Null |

# **Data Dictionary:**

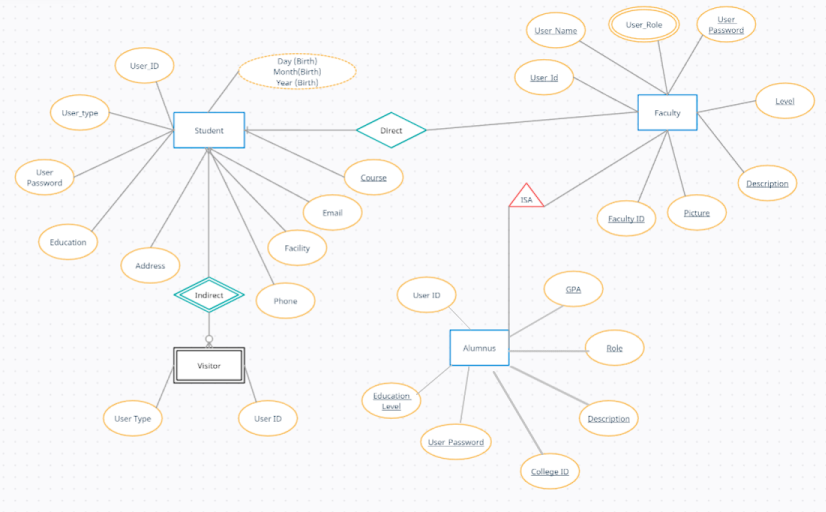
The data dictionary is the proper procedure of explaining data existing in the OAAO platform. The details are:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity | Field | Description | |  |  |
| Student | User ID | Every student will be allotted a unique ID after registering for the app. | | | |
| User type | The type will define the user kind, either active student of alumnus etc. | | | |
| Password | The password will be created | | | |
| Education | The previous education results | | | |
| Day | Day of the month for the birth records | | | |
| Month | Month of the birth | | | |
| Year | Year of the birth | | | |
| Address | Present and permanent address for records | | | |
| Course | Courses enrolled | | | |
| Email | Email details | | | |
| Facility | Facility required such as sports, library, hostel etc | | | |
| Phone | Phone details | | | |
| Year | Enrolled year and left time to graduate | | | |
|  |  |  |  |  |  |
| Visitor | User type | Visitor is Parent/guardian or External Teacher/student etc | | | |
| User ID | Assigned visitor Identity | | | |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| faculty | User ID | The identity locked by administration to distinguish | | | |
| Username | Username set by the users | | | |
| User Pass | Password created to access information | | | |
| User Role | Role such as senior, junior etc | | | |
| Level | Level such as instructor or professor | | | |
| Description | Description such as which subject and course | | | |
| Pic | Picture to identify user | | | |
| Faculty ID | The ID provided by the app to set data records | | | |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| Alumnus | User ID | User Identity for keeping app records | | | |
| Username | Name parsed by user | | | |
| User Pass | Password set to login | | | |
| User Role | Alumnus data such as company serving etc | | | |
| Level | Position in the company | | | |
| Description | Major studied | | | |
| Education level | Educational information such as the higher education, researcher etc | | | |
| GPA | GPA at the time of graduation | | | |
| Roll num | Roll number of classes attended | | | |
| Pic | Picture to identify | | | |
| College ID | College ID for accessing college records from database. | | | |

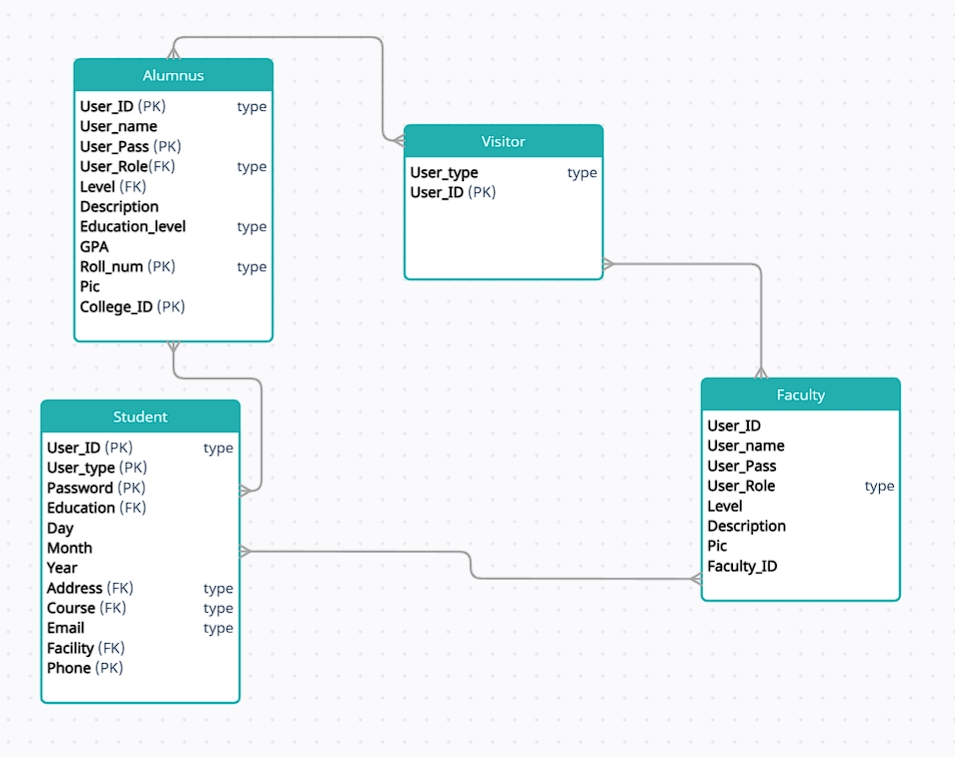
# **Database Description:**

Database description is an important aspect for software design tools to access the planning and strategy of completion of the project. Therefore, the Entity relation Diagram (ERD), Entity Relation mapping and the database pseudo code was used for the OAAO system utilizing MySQL.

The ERD diagram can be seen below:



The ER Mapping can be seen as:



This part presents MySQL code for OAAO database entities with all required data using MySQL Workbench.

create database OAAO;

use OAAO;

**CREATE STUDENT Account**

(

User\_ID decimal(50,0) NoNULL

User\_typevarchar(45) NoNULL

Passworddecimal(50,0) NoNULL

Educationvarchar(45) NoNULL

DaydateNo NULL

MonthdateNo NULL

YeardateNo NULL

Addressvarchar(45)NoNULL

Coursevarchar(45)NoNULL

FOREIGN KEY Emailvarchar(45)NoNULL

Facilityvarchar(45)NoNULL

FOREIGN KEY Phonevarchar(15)NoNULL

);

**CREATE Alumnus Account**

(

User\_ID decimal(50,0) NoNull

User\_namevarchar(35) NoNull

User\_Passint NoNull

User\_Rolevarchar(35) NoNull

Levelvarchar(35) NoNull

Descriptionvarchar(35) NoNull

FOREIGN KEY Education\_level varchar(35) NoNull

GPAfloat NoNull

Roll\_numfloat NoNull

FOREIGN KEY Picvarchar(35) NoNull

FOREIGN KEY College\_IDdecimal(50,0) NoNull);

**CREATE VISTOR Account**

(

User\_typevarchar(45) NoNULL

User\_ID decimal(50,0) NoNULL);

**CREATE Faculty Account**

(

Primary Key User\_ID decimal(50,0) NoNull

User\_name varchar(35) NoNull

User\_Pass int NoNull

User\_Role varchar(35) NoNull

Level varchar(35) NoNull

Description varchar(35) NoNull

Pic varchar(35) NoNull

Faculty\_ID decimal(50,0) NoNull

);

# **Component Design:**

The component design section consists of programming language pseudo code and the natural language processed therefore the logical method is presented below to understand the operational and functional capability of the application. It is to be noted that the specific dev-Ops method was proposed to concern the state-of-the-art conventional technique.

**Common functions**

**Login**

login () {

Enter username and password

If username and password found in database

Then go to the page interface

Else

Show Error message “Incorrect username or password, please try again!”

}

**Forgot password**

forgotPassword () {

Choose a way to get the verification code

If email has chosen

Then enter the email address

If the email found in database

Then send a verification code to the email

Else if phone has chosen

Then enter the phone number

If the phone number found in database

Then send a verification code to the phone number

Else

Show Error message “Invalid, please try again!”

}

**Change password**

changePassword () {

Enter the old password

If the old password found in database

Then enter the new password, confirmed password

If (new password == confirm password)

Update the old password in database

Else

Show Error message “The new password doesn’t match the confirmed password!”

Else

Show Error message “Invalid password!”}

**Edit profile (Student, Faculty, Alumnus)**

editProfile () {

If there is invalid input

Then show Error message “Invalid input, check the format!” Else

Update the fields in database

}

**User functions**

createProfile () {

Enter the user name, user ID, password, profile picture, owner name, entity type, phone number, , email, phone number, password, confirmed password

If some fields missing

Show warning message the missing filed

Else if phone number is not in this format “+#########”

Show warning message “Check the phone number format !”

Else

Save in database

move to the facility page

}

**Access Facility Operations**

AccessingOperations () {

If there is invalid input

Then show Error message “Invalid input, check the format!” Else

Update the fields in database

}

7.2.3 Add new product

addNewOperation () {

if user click on Add button

Enter the operation type, details, timing, description, picture, category

If the user clicked Save button

If some fields missing

Show red asterisk next to the missing filed

Else

Update the operation page

}

# **Detailed System Design:**

The detailed system design provides the details about the buttons and information provided for the functional and operational capability. The low level and subcategory system is provided to understand the working principle of the application system architecture. Also, it is to be noted that the information included also provides the detailed systematic study of the attributes etc.

# **Classification:**

The classification, responsibilities and functionalities and operational Capabilities:

|  |  |  |
| --- | --- | --- |
| Components | Classification | Definitions and Responsibilities |
| Common Functions |  |  |
| Login | Functions | This the function will allow the user to access their account by entering the email and password |
| Forget Password | Functions | This function will check if the email entered by the user matches any of the emails stored in the database |
| Change Password | Functions | This function will allow the user to retrieve the password by e- mail or phone number. |
| Edit Profile | Functions | This function allows to modify the profile information and control the account |
| User Functions |  |  |
| Create Account | Functions | This function allows the user to create the profile by all information needed for the creation |
| Edit account information | Functions | The user can manage their profile information |
| Delete account information | Functions | The user can delete their profile information |
| Add new operation | Functions | The user can add operation/facility information |
| Delete Operation | Functions | The user can delete operation/facility information |
| Admin Functions |  |  |
| Create Account | Functions | This function allows user to register in the platform. The user must provide first name, last name, password, email, phone number, birth date, address, and gender. The customer must agree on the terms of use and have read the privacy policy |
| Delete account | Functions | This function will allow user to delete someone’s profile |

After the main system, the sub system clarifies the classification based on the sub system and compositional information:

|  |  |  |  |
| --- | --- | --- | --- |
| Components | Constraints | Pre-Condition | Post Condition |
| Common Functions |  |  |  |
| Login | The user must first create an account | Enter email/phone number and password | Check if the information is valid |
| Forgot Password | The user must first create an account | Enter phone number/ email | Send code to the email/phone number and reset the new password. |
| Change Password | The password must be 8 characters long including at least: • 1 Uppercase letter • 1 Lowercase letters • Numbers | Enter the new password. | The new password will be store in the database of the user |
| Edit Profile | The data must be valid, and all the fields must be filled | Enter personal information | The data will be store in the database of the customer |
| Edit Operation | None | Edit on the operation/facility information | Store the update in the database of the operations/facility |
| Delete Operation | None | Remove on the operation/facility information | Delete and update in the database of the operations/facility |