**Faculty Workload and Resource Allocation System**



Session: 2023 – 2027

**Submitted** **by:**

Marsad Manzoor 2023R2022-CS-203

**Supervised** **by:**

Mr. Samyan Qayyam Wahla

Department of Computer Science **University** **of** **Engineering** **and** **Technology**

**Lahore** **Pakistan**

**Contents**

[List of Figures 4](#_Toc193486280)

[List of Tables 4](#_Toc193486281)

[1.1 Project Overview 1](#_Toc193486282)

[1.1.1 Summary 1](#_Toc193486283)

[1.1.2 Motivation 1](#_Toc193486284)

[1.1.3 Objectives 2](#_Toc193486285)

[1.1.4 Target Audience 2](#_Toc193486286)

[1.1.5 Stakeholders 2](#_Toc193486287)

[1.1.6 Operational Details 3](#_Toc193486288)

[1.2 Use Cases 4](#_Toc193486289)

[1.2.1 User Authentication and Role Management 4](#_Toc193486290)

[1.2.2 Faculty Profile Management 4](#_Toc193486291)

[1.2.3 Workload Assigement 4](#_Toc193486292)

[1.2.4 Resource Request Submission 5](#_Toc193486293)

[1.2.5 Resource Request and Approval 5](#_Toc193486294)

[1.2.6 Class and Lab Space Allocation 5](#_Toc193486295)

[1.2.7 Administrative Task Management 5](#_Toc193486296)

[1.2.8 Report Generation 5](#_Toc193486297)

[1.2.9 Exception Handling and Notification 5](#_Toc193486298)

[1.2.10 System Security and Data Backup 6](#_Toc193486299)

[1.3 Wire Frames 6](#_Toc193486300)

[1.3.1 Welcome Page 6](#_Toc193486301)

[1.3.2 Sign Up Form 7](#_Toc193486302)

[1.3.3 Sign In From 8](#_Toc193486303)

[1.3.4 Faculty Form 9](#_Toc193486304)

[1.3.5 Track Requests Form 11](#_Toc193486305)

[1.3.6 Admin Form 12](#_Toc193486306)

[1.3.7 Admin Update Faculty Form 13](#_Toc193486307)

[1.3.8 Head Form 14](#_Toc193486308)

[1.3.9 Faculty Requests To Head Form 15](#_Toc193486309)

[1.3.10 Admin Course Allocation Form 16](#_Toc193486310)

[1.3.11 Admin Course View Form 17](#_Toc193486311)

[1.3.12 Admin Course Allocation Form 17](#_Toc193486312)

[1.3.13 Admin Create Course Form 19](#_Toc193486313)

[1.4 Database Details 21](#_Toc193486314)

[1.4.1 Database Design 21](#_Toc193486315)

[1.4.2 Database Tables 21](#_Toc193486316)

[1.5 Implementation 26](#_Toc193486317)

[1.6 Project Details 31](#_Toc193486318)

i

*Contents* ii

1.6.1 Languages Used . . . . . . . . . . . . . . . . . . . . . . . . . 45

# List of Figures

|  |  |
| --- | --- |
| * 1. Wire frame of Welcome Form . . . . . . . . . . . . . . . . . . . . . . | 6 |
| 1.2 Wire frame of Sign Up Form . . . . . . . . . . . . . . . . . . . . . . | 7 |
| 1.3 Wire frame of User Login Form . . . . . . . . . . . . . . . . . . . | 8 |
| 1.4 Wire frame of faculty Form . . . . . . . . . . . . . . . . . . . . . . . | 10 |
| 1.5 Wire frame of Group Track Requests Form . . . . . . . . . . | 11 |
| 1.6 Wire frame of Admin Form . ………….. . . . . . . . . . . . . . . . . | 12 |
| 1.7 Wire frame of Update Faculty Form . . . . . . . . . . . . . . . . . | 13 |
| 1.8 Wire frame of Head Form . . . . . . . . . . . . . . . . . . . . | 14 |
| 1.9 Wire frame of Faculty Request Form . . . . . . . . . . . . . . . . | 15 |

iii

# List of Tables

|  |  |
| --- | --- |
| 1.1 Welcome Form Component Details . . . . . . . . . . . . . . . . . . . | 6 |
| 61.2 SignUp Form Component Details . . . . . . . . . . . . . . . . . . . | 7 |
| 1.3 User Login Form Component Details . . . . . . . . . . . . . . . . . . . | 9 |
| 1.4 Faculty Form Component Details . . . . . . . . . . . . . . . . . . . . | 10 |
| 1.5 Track Requests Form Component Details . . . . . . . . . . . . . . . | 11 |
| 1.6 Admin Form Component Details . . . . . . . . . . . . . . . | 12 |
| 1.7 Update Faculty Form Component Details . . . . . . . . . . . . . . | 13 |
| 1.8 Head Main Form Component Detail. . . . . . . . . . . . . . . . . | 14 |
| 1.9 Faculty Requests Form Component Details . . . . . . . . . . . . . | 15 |
|  |  |

iv

**Faculty Workload and Resource Allocation System**

## 1.1 Project Overview

### 1.1.1 Summary

The Faculty Workload and Resource Allocation System is a desktop-based application developed to automate and streamline faculty workload distribution and resource management in the Department of Computer Science, UET Lahore. The current manual system, reliant on spreadsheets and emails, leads to data inconsistencies, workload imbalances, inefficient tracking of faculty duties, and delays in processing resource requests such as classrooms, lab spaces, and consumables. Additionally, the manual preparation of reports for accreditation bodies like NCEAC and HEC is time-consuming and prone to errors. To address these challenges, the proposed system provides a centralized platform that ensures equitable workload distribution, efficient resource allocation, and secure role-based access control (RBAC) for department heads, faculty members, and administrative staff. Key functionalities include faculty profile management, workload assignment for teaching and research, resource request handling, real-time tracking of request approvals, and automated report generation. The system is designed as a Windows Form Application using C# and a relational database, featuring a secure login system with hashed passwords, an intuitive user interface, and efficient database connectivity. By eliminating manual inefficiencies, the system enhances accuracy, transparency, and decision-making for departmental planning and faculty workload management.

### 1.1.2 Motivation

Hand-managing the faculty workload and resource allocation with the help of spreadsheets and emails has caused various inefficiencies within the Department of Computer Science, UET Lahore. It leads to a lack of structure and creates imbalance in workload with some of the faculty members bearing heavy workloads while others bear less. Further, monitoring the faculty responsibilities such as teaching, research supervision, and administrative work is not possible, which leads to mismanagement and conflicts during scheduling. Professors often ask for classrooms, lab facilities, board markers, and other consumables, but because the approval process is unstructured, these requests get lost or delayed.

Another significant challenge is the manual generation of reports to departmental planning and accrediting agencies such as NCEAC and HEC, which is a lengthy process with scope for errors. Lack of role-based authentication also creates security issues, as unauthorized access to faculty information may result in alterations or abuse of vital data.

For these challenges to be addressed, there is a need for an automated Faculty Workload and Resource Allocation System. This system will provide fair workload allocation, offer an efficient process for faculty resource requests, and prepare automatic reports, enhancing efficiency, accuracy, and transparency in departmental operations. By eliminating manual processes with a centralized, database-based solution, the system will increase productivity, decrease administrative load, and enable well-informed decision-making, ultimately leading to enhanced faculty management and institutional effectiveness.

### 1.1.3 Objectives

1. To develop a system that is systematic and comfortable.
2. To develop a system which is error free
3. To test the reliability of the system
4. To develop an automated and structured system for managing faculty workload distribution and resource allocation, ensuring equitable workload assignment and efficient handling of faculty requests

### 1.1.4 Target Audience

Educational institutes.

### 1.1.5 Stakeholders

**Department Head:** Assigns workload, approves faculty requests, and manages resource allocation.

**Faculty Members:** View assigned courses, submit resource requests, and track their status.

**Administrative Staff:** Update faculty profiles, manage course allocations, and process faculty requests.

### 1.1.6 Operational Details

The Faculty Workload and Resource Allocation System is a desktop-based Windows Form Application, developed using C# and a relational database, to streamline faculty workload distribution and resource management in the Department of Computer Science, UET Lahore. The system is designed to automate the assignment of faculty responsibilities, track resource requests, and generate detailed reports for departmental planning and accreditation purposes.

The system will feature a secure login mechanism with role-based access control (RBAC) to ensure that users can only access the functionalities relevant to their roles. The Department Head will have the authority to assign workload, approve resource requests, and oversee faculty allocations, while faculty members will be able to view their assigned courses, research duties, and submit requests for resources such as classrooms, lab spaces, and consumables. Additionally, administrative staff will handle faculty profile management, course allocations, and request processing.

The workload assignment module will allow the department to distribute teaching assignments, research supervision, and administrative duties among faculty members while ensuring workload balance. The resource allocation module will enable faculty members to submit requests for necessary resources, which can then be approved, rejected, or modified by the department head. The system will also maintain an inventory of consumables to ensure timely procurement.

To improve efficiency, the system will automatically generate reports on faculty workload distribution, resource allocation status, and request histories, which will be used for departmental planning and accreditation reports for bodies like NCEAC and HEC. The system will be implemented with a user-friendly interface, ensuring ease of use for faculty members and administrators, while also maintaining data security and integrity through database-driven authentication and access control mechanisms..

## 1.2 Use Cases

Following are the use cases implemented in the system

1. User Authentication and Role Management
2. Faculty Profile Management
3. Work Load Assignment
4. Resource Request Submission
5. Resource Request and Approval
6. Class and Lab Space Allocation
7. Administrative Task Management
8. Report Generation
9. Exception Handling and Notification
10. System Security and Data Backup

### 1.2.1 User Authentication and Role Management

The system allows faculty members, department heads, and administrative staff to log in securely using role-based access control (RBAC). Users can reset passwords and recover accounts if needed.

### 1.2.2 Faculty Profile Management

The administrative staff can add, update, and delete faculty profiles. The system stores faculty details such as name, designation, research interests, total available teaching hours, and assigned courses.

### 1.2.3 Workload Assigement

The department head can assign courses, final year projects (FYPs), research supervision, and administrative responsibilities to faculty members. The system ensures that the workload is evenly distributed among faculty members.

### 1.2.4 Resource Request Submission

Faculty members can submit requests for classrooms, lab spaces, board markers, stationery, and consumables. Each request includes details such as priority level, justification, and requested quantity.

### 1.2.5 Resource Request and Approval

The department head can approve, reject, or modify faculty resource requests. Faculty members can track their request status in real-time to check whether it has been approved, rejected, or is still pending

### 1.2.6 Class and Lab Space Allocation

The department head assigns classrooms and lab spaces to faculty members based on their teaching schedules. The system prevents scheduling conflicts by checking for overlapping reservations.

### 1.2.7 Administrative Task Management

The department can assign, track, and manage administrative duties such as committee roles, event coordination, and faculty meetings. Faculty members can view and update their assigned responsibilities.

### 1.2.8 Report Generation

The system generates automated reports for faculty workload distribution, resource allocation status, and pending faculty requests. Reports can be exported in PDF format for departmental planning and accreditation purposes (NCEAC, HEC).

### 1.2.9 Exception Handling and Notification

The system provides error handling mechanisms to prevent system failures. Faculty members receive notifications regarding approved, rejected, or pending requests. Errors such as invalid login attempts or resource request conflicts are displayed as user-friendly messages..

### 1.2.10 System Security and Data Backup

The system ensures secure faculty records storage using encryption techniques and restricted access. Periodic data backups are performed to prevent data loss and ensure system reliability.

## 1.3 Wire Frames

### 1.3.1 Welcome Page

This page allows users to sign in, sign up, or exit the application. The Sign In button authenticates users, the Sign Up button redirects to registration, and the Back button closes the program.

**

Figure 1.1: Wire frame of Welcome Form

Table 1.1: Welcome Form Component Details

|  |  |  |
| --- | --- | --- |
| **Component**  **Name** | **Component**  **Type** | **Description** |
| Title | Label | To show to label of the Welcome Form |
| Sign Up | Button | To redirect the user to the Sign Up Page |
| Sign In | Button | The Sign In button authenticates users |

### 1.3.2 Sign Up Form

The Sign-Up Page allows new users to create an account by entering their details such as name, email, and password. It includes a sign-up button to submit the registration and a back button to return to the previous screen.

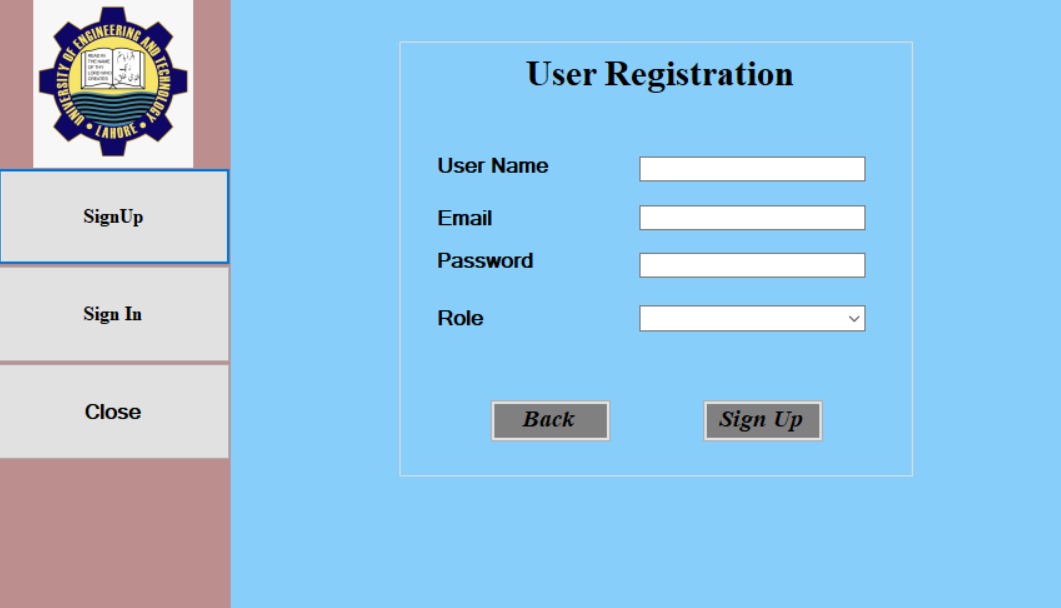


Figure 1.2: Wire frame of Sign Up Page

Table 1.2: Sign Up Form Component Details

|  |  |  |
| --- | --- | --- |
| **Component**  **Name** | **Component**  **Type** | **Description** |
| Username | Label | To show to label of Username |
| Email | Label | To show the label of Email ID |
| Password | Label | To show the label of Password |
| Role | Drop Down | To show the selective number of the roles |
| Username TextBox | TextBox | To input the username |
| Email  TextBox | TextBox | To input the email |
| Password TextBox | TextBox | To input the password |
| Role ComboBox | ComboBox | To add the role |
| Back | Button | To back to the main page |
| Sign Up | Button | To successfully add the user |

### 1.3.3 Sign In From

Sign-In Form allows users to enter their email and password to access the system. It includes input fields for credentials and a login button to authenticate users.

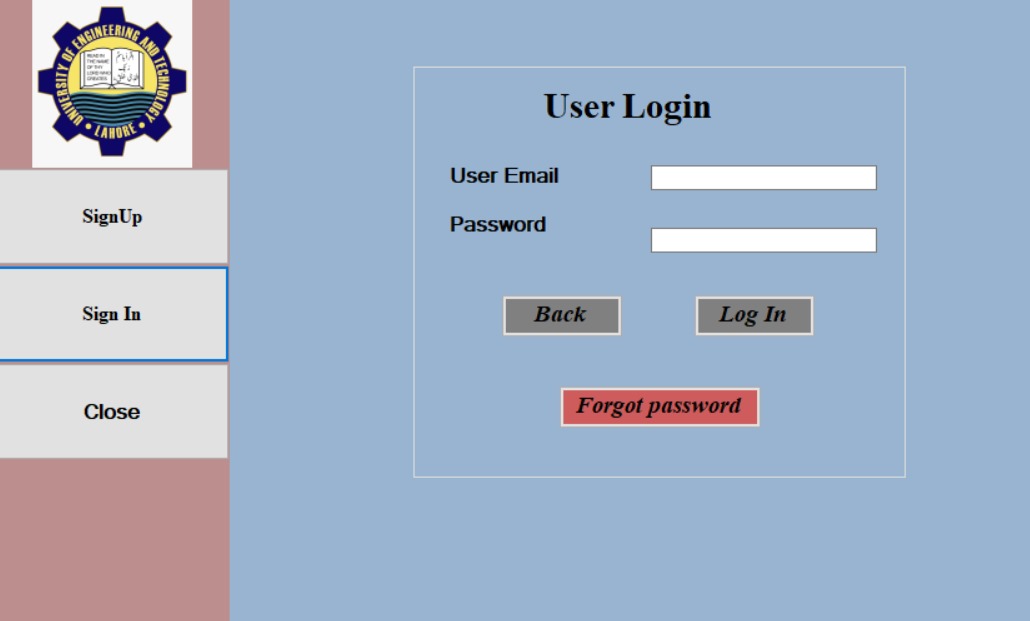


Figure 1.3: Wire frame of User Login Form

Table 1.3: User Login Form Component Details

|  |  |  |
| --- | --- | --- |
| **Component**  **Name** | **Component**  **Type** | **Description** |
| User Login | Label | To show to label of User Login |
| User Email | Label | To show the label of the User Email |
| Password | Label | To show the label of the label |
| User Email TextBox | TextBox | To input the email of the user |
| Password TextBox | TextBox | To input the password of the user |
| Back | Button | To redirect to the main page |
| Login | Button | To redirect the user to the specific form against their role |
| Forgot the password | Button | To get the user to page where they can recover its password |

### 1.3.4 Faculty Form

The Faculty Window provides faculty members with options to view courses, request resources, and track their requests. It features a sidebar with navigation buttons and interactive sections for submitting and managing requests.

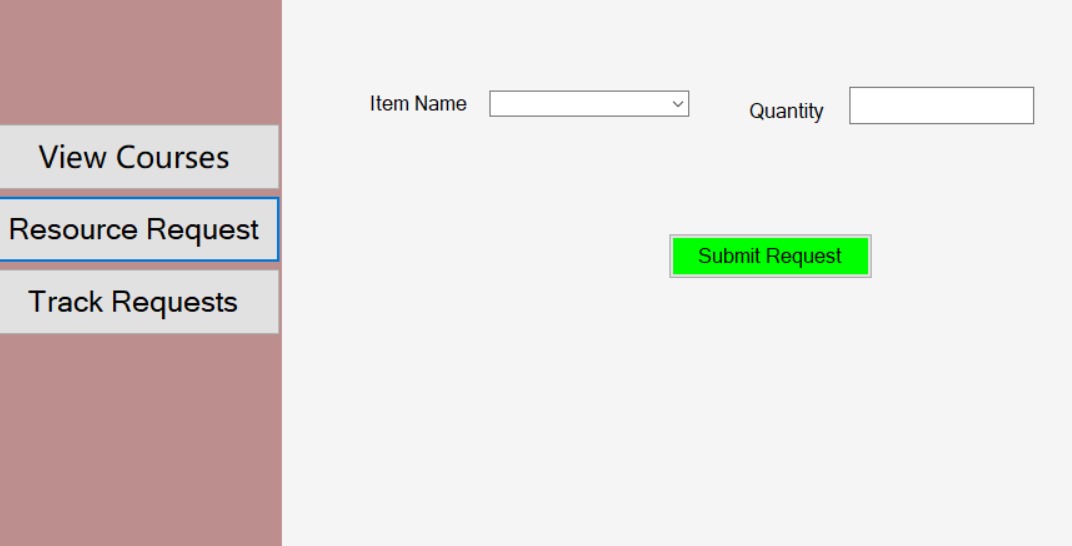


Figure 1.4: Wire frame of Faculty Form and Request Form

Table 1.4: Faculty Form Component Details

|  |  |  |
| --- | --- | --- |
| **Component**  **Name** | **Component**  **Type** | **Description** |
| View Courses | Button | To show the courses that are assigned to the specific faculty member |
| Request Resource | Button | To request the Head of the departement for specific resource |
| Track the Request | Button | To redirect the user to form so they track they their requests |
| Item Name | ComboBox | To pick the item from the available item |
| Quantity | TextBox | To select the Quantity |
| Submit Request | Button | To submit the request for the resource required |

### 1.3.5 Track Requests Form

The Track Requests Form allows faculty members to view the status of their submitted resource requests. It includes a "Load My Requests" button, which retrieves and displays the faculty member's previous requests in the gray display area. The left-side navigation panel provides quick access to other features like viewing assigned courses and making new resource requests

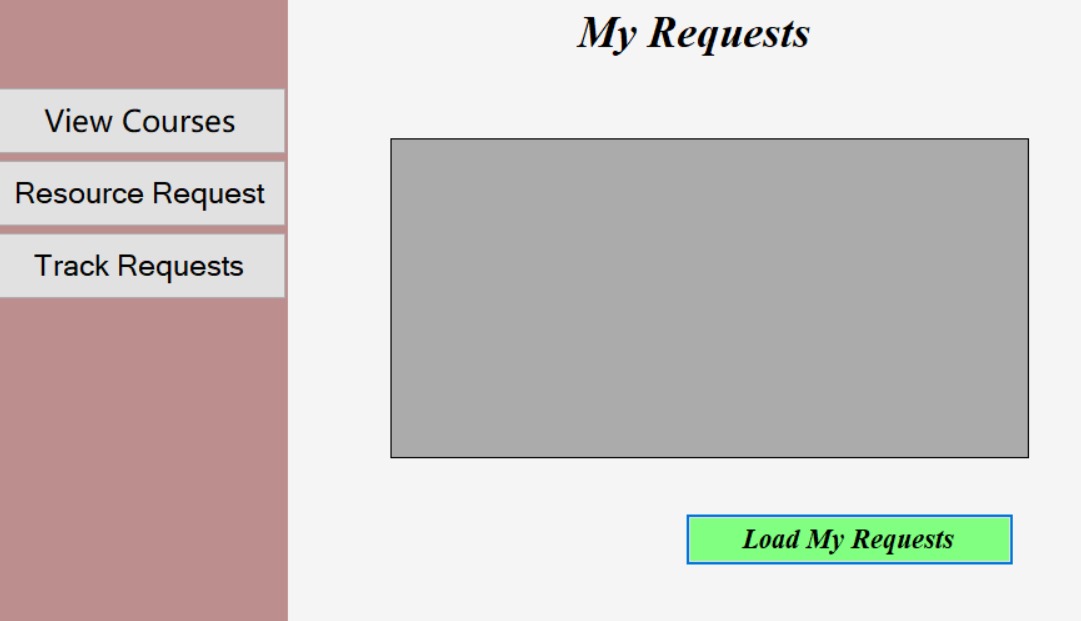


Figure 1.5: Wire frame of Track Requests Form

Table 1.5: Track Requests Form Component Details

|  |  |  |
| --- | --- | --- |
| **Component**  **Name** | **Component**  **Type** | **Description** |
| Title | Label | To show to label of Track Requests Form |
| Load My Requests | Button | To Show the requests of the Faculty member to the Head |
| List My requests | DataGridView | To show the Data Grid View of the Requests |
| Group ID | Label | To show the label to add group ID |

### 1.3.6 Admin Form

The Admin Form allows administrators to manage faculty and course allocations efficiently. It includes options to update faculty details, allocate courses to faculty members, and handle faculty requests. The "Update Faculty" button enables modifications to faculty information, while "Course Allocation" assigns courses based on expertise and availability. The "Faculty Requests" section helps administrators review and process faculty-submitted requests. This form provides a streamlined way to oversee faculty-related operations.



Figure 1.6: Wire frame of Admin Form

Table 1.6: Admin Form Component Details

|  |  |  |
| --- | --- | --- |
| **Component**  **Name** | **Component**  **Type** | **Description** |
| UpdateFaculty | Button | To update the Faculty in the department |
| Course Allocation | Button | To assigned the course to the faculty |
| Faculty Requests | Button | To check the faculty requests |

### 

### 1.3.7 Admin Update Faculty Form

This Update Faculty Form allows the admin to manage faculty records efficiently. It includes three main options: "Add Faculty" to register new faculty members, "Update Faculty" to modify existing faculty details, and "Delete Faculty" to remove faculty from the system. These options help streamline faculty management within the institution.

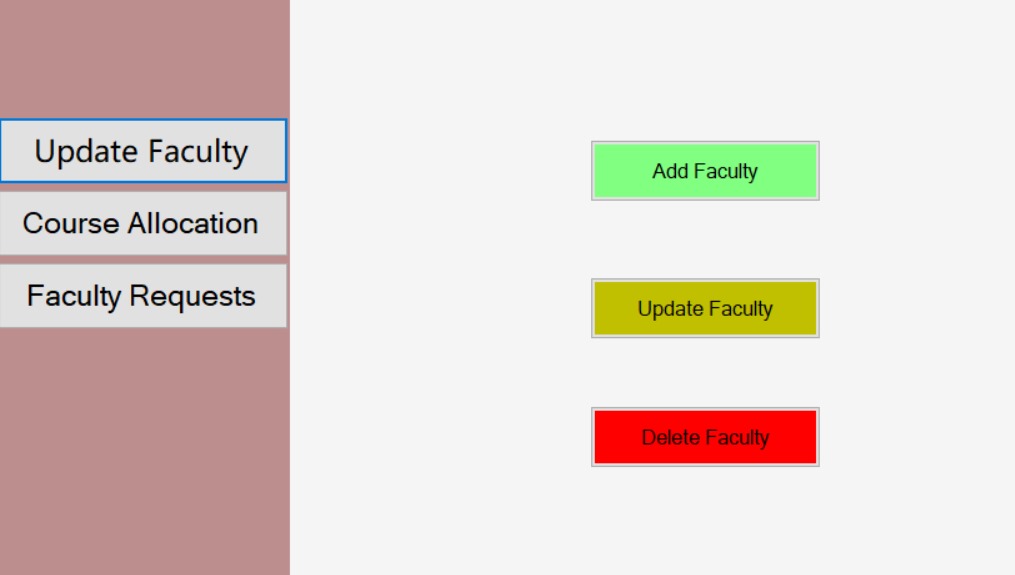


Figure 1.7: Wire frame of Update Faculty Form

Table 1.7: Update Faculty Form Component Details

|  |  |  |
| --- | --- | --- |
| **Component**  **Name** | **Component**  **Type** | **Description** |
| Add Faculty | Button | To Add the new Faculty |
| Update Faculty | Button | To update the exciting Faculty |
| Delete Faculty | Button | To delete the Faculty |

### 

### 1.3.8 Head Form

The **Head Form** is designed for the **Head of the Department (HOD)** to manage faculty and departmental tasks. It has three main options: **Assign Workload** to distribute tasks among faculty, **Handle Requests** to manage faculty requests, and **Add Resources** to add departmental resources. The interface has a side panel for navigation and a large workspace to display selected functions, making it easy for the HOD to manage responsibilities efficiently

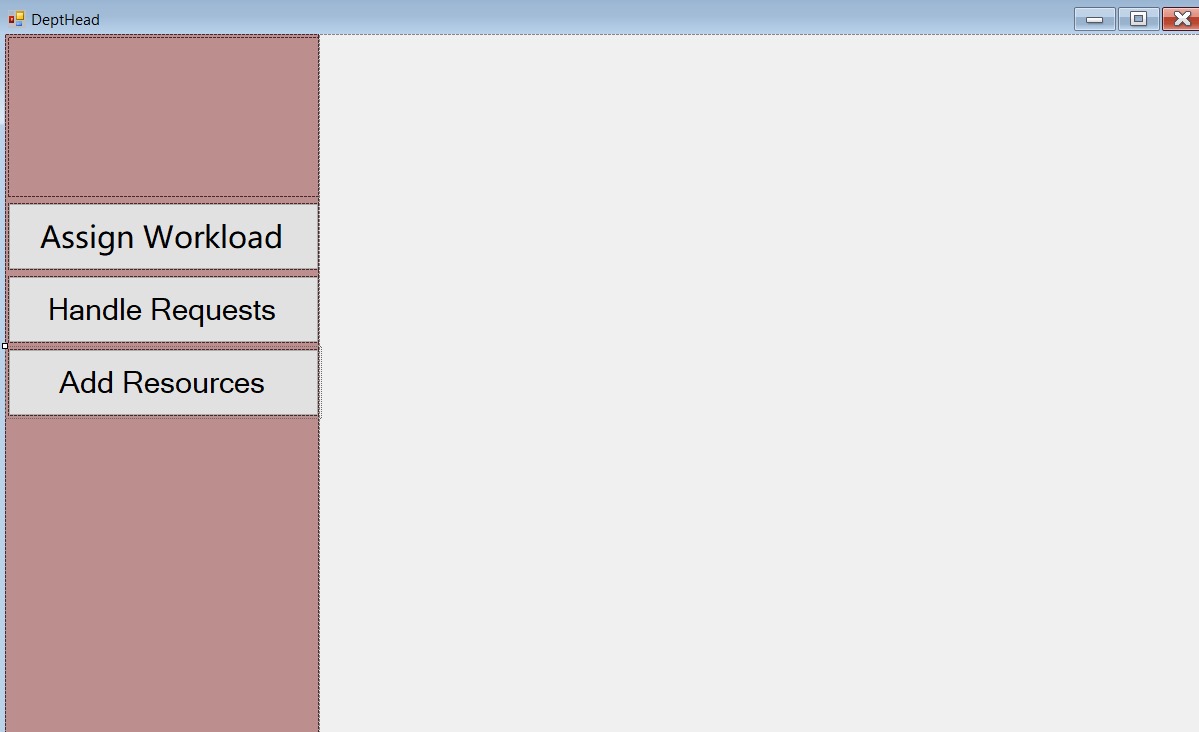


Figure 1.8: Wire frame of Head Form

Table 1.8: Head main Form Component Details

|  |  |  |
| --- | --- | --- |
| **Component**  **Name** | **Component**  **Type** | **Description** |
| Assign Workload | Button | To assign a work to a specific teacher or faculty member |
| Handle Requests | Button | To Handle the requests form the faculty |
| Add Resource | Button | To add the resources that would be available for the administration and faculty |

### 1.3.9 Faculty Requests To Head Form

The Faculty to Head Requests interface allows the Head of the Department (HOD) to view and manage requests submitted by faculty members. The form displays a data grid view listing faculty requests, including the faculty name, item name, quantity, and request status. The "Load Requests" button fetches and updates the list of requests. The HOD can use this interface to review and process faculty needs efficiently.

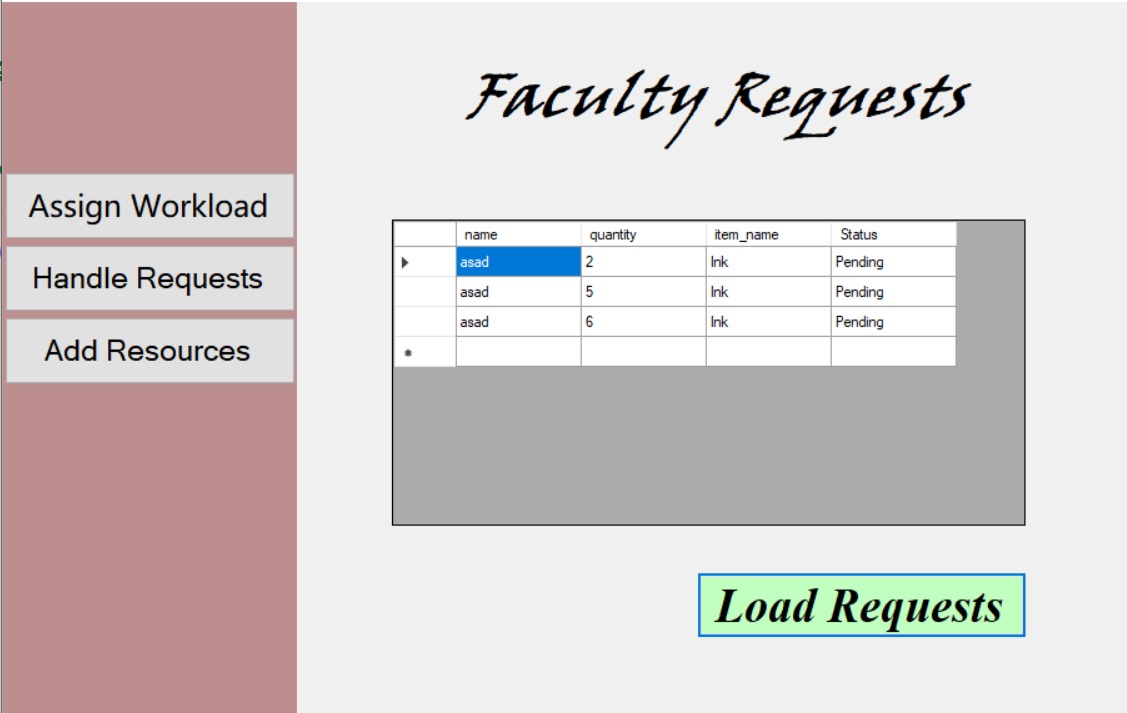


Figure 1.9: Wire frame of Faculty Requests Form

Table 1.9: Faculty Requests Form Component Details

|  |  |  |
| --- | --- | --- |
| **Component**  **Name** | **Component**  **Type** | **Description** |
| Faculty Requests | Label | To show to label of faculty requests |
| Load Requests | Button | The button that load all the requests from faculty |
| Faculty Requests Viewer | Data Grid View | To view the requests from the faculty |

### 1.3.10 Admin Course Allocation Form

The Admin Course Allocation interface is designed to help administrators manage faculty and course assignments efficiently. On the left panel, there are navigation options to update faculty, allocate courses, and handle faculty requests. The main section contains buttons for different actions. The "Create Course" button allows the admin to add new courses to the system, while the "Allocate Course" button is used to assign courses to faculty members. The "View Courses" button provides a list of available courses and their details. This interface streamlines the process of course management within the institution.



Figure 1.10: Wire frame of Course Allocation Form

Table 1.10: Report Form Component Details

|  |  |  |
| --- | --- | --- |
| **Component**  **Name** | **Component**  **Type** | **Description** |
| Create Course | Button | To create a course |
| Allocate | Button | To allocate a course |
| View Course | Button | To view the courses |

## 1.3.11 Admin Course View Form

The Admin Course View Form is designed to provide administrators with an overview of allocated courses. The left panel contains navigation options, including updating faculty, course allocation, and faculty requests. The main section displays a table with course details, such as the academic year, term, credit hours, course type, course name, total teaching hours, faculty contact, and faculty name. The interface also features a "Refresh" button, allowing the admin to update the displayed data to reflect any recent changes. The design helps in managing and monitoring faculty course assignments efficiently



Figure 1.10: Wire frame of Course viewer Form

Table 1.10: Course Viewer Form Component Details

|  |  |  |
| --- | --- | --- |
| **Component**  **Name** | **Component**  **Type** | **Description** |
| Refresh | Button | To refresh the courses |
| Course Viewer | Data Grid Viewer | To view the all courses |

## 1.3.12 Admin Course Allocation Form

The Admin Course Allocation Form provides an interface for administrators to assign courses to faculty members efficiently. On the left side, a table displays the list of available courses, including details such as credit hours, course type, course name, and course ID. On the right side, another table lists faculty members along with their faculty ID, name, email, contact information, and research area. These tables enable the administrator to view and select the appropriate course and faculty member for allocation. Below the tables, a dropdown menu is available, likely for selecting additional allocation criteria. The form also features a prominent "Allocate" button, which finalizes the course assignment process once the selections are made. The interface is structured to ensure a smooth and organized workflow for course allocation.

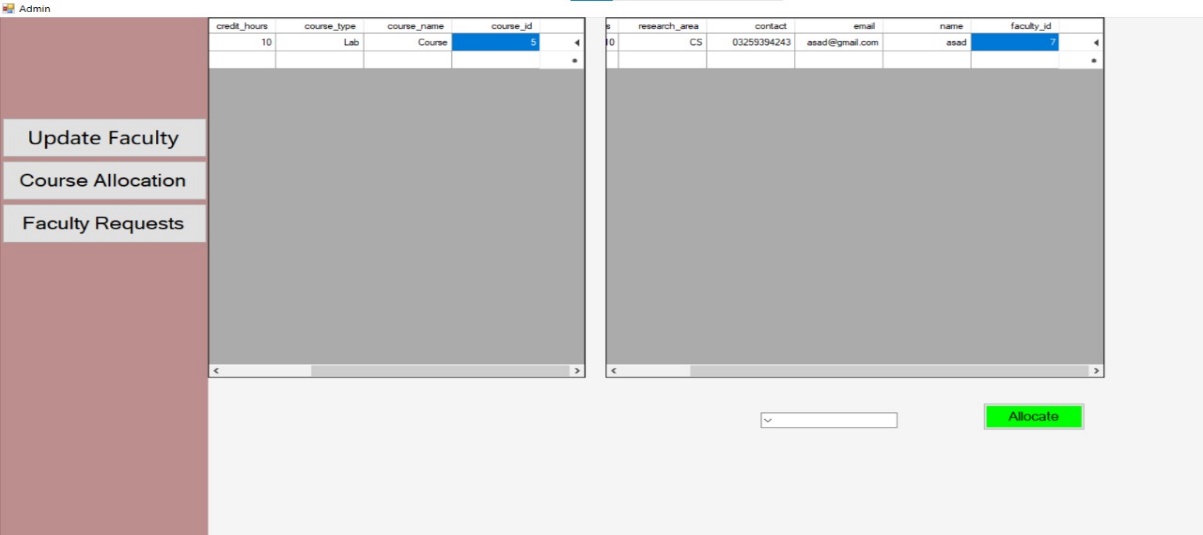


Figure 1.10: Wire frame of Course Allocation Form

Table 1.10: Course Allocation Form Component Details

|  |  |  |
| --- | --- | --- |
| **Component**  **Name** | **Component**  **Type** | **Description** |
| Allocate | Button | To Allocate the course to a faculty member |
| Faculty Grid View | Data Grid Viewer | To view the all Faculty member |
| Course Grid View | Data Grid View | To view the all courses available |

## 1.3.13 Admin Create Course Form

The Admin Course Allocation Form provides an interface for administrators to assign courses to faculty members efficiently. On the left side, a table displays the list of available courses, including details such as credit hours, course type, course name, and course ID. On the right side, another table lists faculty members along with their faculty ID, name, email, contact information, and research area. These tables enable the administrator to view and select the appropriate course and faculty member for allocation. Below the tables, a dropdown menu is available, likely for selecting additional allocation criteria. The form also features a prominent "Allocate" button, which finalizes the course assignment process once the selections are made. The interface is structured to ensure a smooth and organized workflow for course allocation.

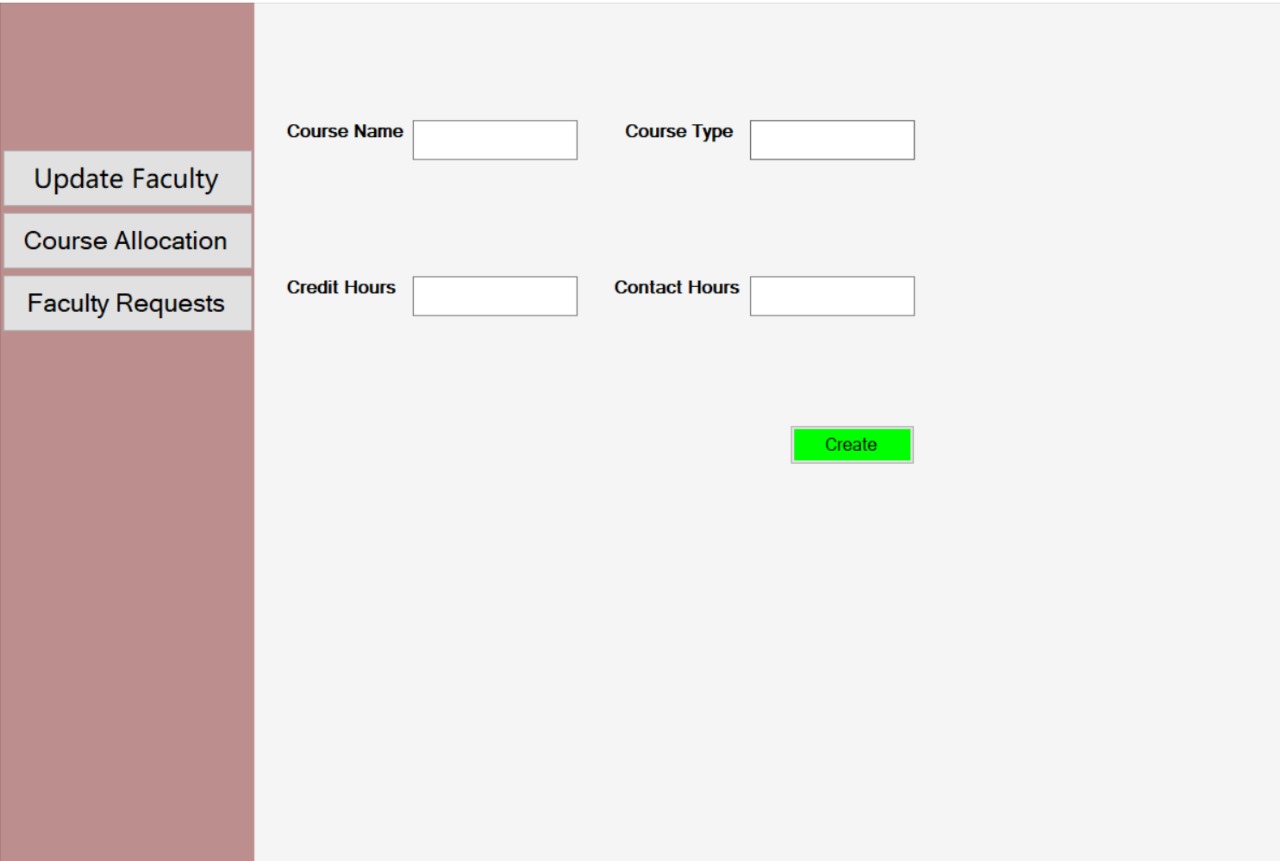


Figure 1.10: Wire frame of Create Course Form

Table 1.10: Create Course Form Component Details

|  |  |  |
| --- | --- | --- |
| **Component**  **Name** | **Component**  **Type** | **Description** |
| Course Name | Lable | To show the Course Name title |
| Course Type | Label | To show the Course Type title |
| Credit Hours | Label | To show the credit Hours title |
| Contact Hours | Label | To show the Contact Hours title |
| Course Name Text Box | Text Box | To write the Course Name |
| Course Type | Text Box | To write the Course Type |
| Credit Hours | Text Box | To write the Credit Hours |
| Contact Hours | Text Box | To write the Contact Hours |
| Create | Button | To create the Course |

## 1.4 Database Details

### 1.4.1 Database Design

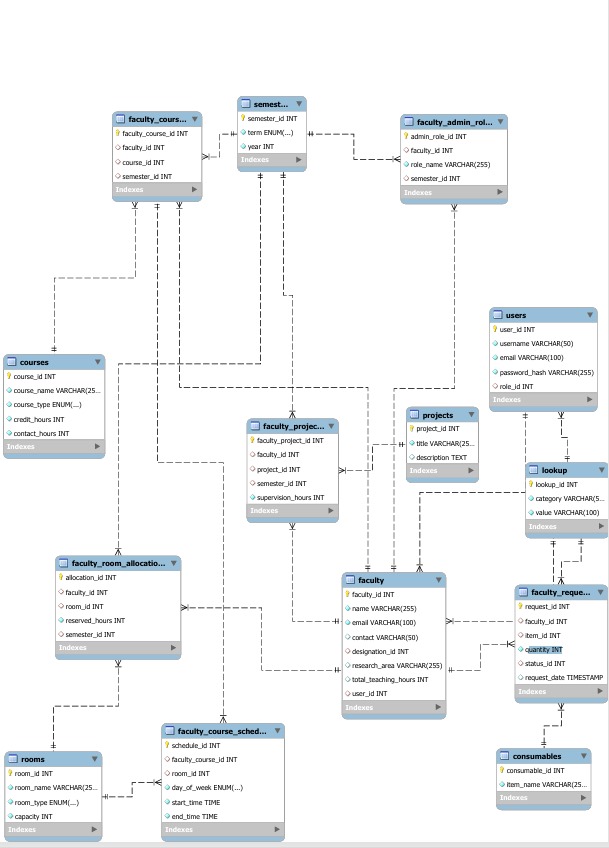


Figure 1.11: Database Design

### 1.4.2 Database Tables

Following are the tables in the database and the detail:

1. Consumables
2. Courses
3. Faculty Table
4. Faculty Admin Roles
5. Faculty Course Assignment Table
6. Users Tabel
7. Semester Table
8. Faculty Room Allocation Table
9. Faculty Room Allocation
10. Rooms Tables
11. Lookup
12. Users Table
13. Semester Table

#### 1. Consumables

The Consumables table stores information about available consumable resources. Each resource is identified by consumable\_id, which is an auto-incremented primary key. The item\_name field provides the name of the resource. Consumables are added to the system by administrators for faculty use

#### 2. Courses

The Courses table stores information about all courses offered by the department. Each course is uniquely identified by course\_id, which is an auto-incremented primary key. The course\_name field holds the title of the course, and course\_type categorizes the course as either Theory or Lab. The credit\_hours field defines the academic weight of the course, while contact\_hours records the number of weekly instructional hours. New courses are added by the administrator when defining the curriculum.

#### 3. Faculty Table

The Faculty table contains essential details about faculty members, including their professional and contact information. The primary key is faculty\_id, which is auto-incremented to ensure uniqueness. The name field stores the full name of the faculty member, while email serves as a unique identifier for communication purposes. The contact field is optional and stores phone numbers. The designation field indicates the faculty member's position, such as Professor or Lecturer. The research\_area field holds information about their academic specialization. Additionally, total\_teaching\_hours records the total workload assigned to a faculty member, and the user\_id column links faculty members to their authentication details in the users table. Faculty details are inserted when they register in the system

#### 4. Faculty Admin Roles

The Faculty Admin Role table assigns administrative roles to faculty members. The primary key is admin\_role\_id, which is auto-incremented for uniqueness. The faculty\_id field references the faculty table, identifying which faculty member holds an administrative position. The role\_name field describes the assigned role, such as Head of Department or Coordinator. The semester\_id field ensures that roles are linked to specific academic terms. An entry is created whenever a faculty member is given an administrative responsibility.

#### 5. Faculty Course Assigement Table

The Faculty Course Assignment table manages the assignment of courses to faculty members. Each entry is uniquely identified by faculty\_course\_id, which is an auto-incremented primary key. The faculty\_id and course\_id fields reference the faculty and courses tables, ensuring that only valid faculty members are assigned to existing courses. The semester\_id field links assignments to a specific academic semester. Whenever a faculty member is assigned a course, a corresponding entry is created in this table.

#### 6. Users Table

The **Users** table stores authentication details for all users in the system, such as faculty members and administrators. Each user is assigned a unique user\_id, which is an auto-incremented primary key. The username and email fields are mandatory and unique, ensuring that each user has a distinct identity. The password\_hash field securely stores encrypted passwords. The role\_id is a foreign key referencing the lookup table, which categorizes users into roles such as Admin and Faculty. Each user is stored with a hashed password to ensure security

#### 7. Semester Table

The Semester table keeps track of academic terms. Each semester is identified by semester\_id, which is an auto-incremented primary key. The term field categorizes the semester as either Fall or Spring, and the year field records the academic year. A new entry is created at the beginning of each semester to ensure proper tracking of faculty workloads and course schedules.

#### 8. Faculty Room Allocation Table

The Faculty Room Allocation table manages room assignments for faculty members. Each allocation entry has a unique allocation\_id, which is an auto-incremented primary key. The faculty\_id and room\_id fields reference the faculty and rooms tables, ensuring that only valid faculty members are assigned rooms. The reserved\_hours field records the number of hours a faculty member is allocated a specific room. The semester\_id field links the allocation to a particular academic semester. An entry is added whenever a faculty member reserves a room for lectures, research, or administrative purposes..

#### 9. Rooms Table

The Rooms table stores information about all rooms available for lectures, labs, and faculty offices. Each room is uniquely identified by room\_id, which is an auto-incremented primary key. The room\_name field holds a unique name or number assigned to each room, while the room\_type field categorizes it as a classroom, lab, or office. Rooms are predefined in the system to facilitate scheduling and allocations.

#### 10. Projects Table

The Projects table stores details of research and development projects undertaken by faculty members. Each project is uniquely identified by project\_id, which is an auto-incremented primary key. The title field provides the name of the project, while the description field elaborates on its objectives and scope. New projects are added by faculty members as part of their research initiatives.

#### 11. Lookup

The Lookup table serves as a general repository for predefined values used across the database. Each entry is identified by lookup\_id, which is an auto-incremented primary key. The category field classifies the type of lookup value, such as 'Course Type' or 'User Role', while the value field stores the corresponding data (e.g., 'Theory', 'Lab', 'Admin'). This table standardizes the data used in various other tables.

12. **Users Table**

The **Users** table stores authentication details for all users in the system, such as faculty members and administrators. Each user is assigned a unique user\_id, which is an auto-incremented primary key. The username and email fields are mandatory and unique, ensuring that each user has a distinct identity. The password\_hash field securely stores encrypted passwords. The role\_id is a foreign key referencing the lookup table, which categorizes users into roles such as Admin and Faculty. Each user is stored with a hashed password to ensure security

13. **Semesters** **Table**

The **Semester** table keeps track of academic terms. Each semester is identified by semester\_id, which is an auto-incremented primary key. The term field categorizes the semester as either Fall or Spring, and the year field records the academic year. A new entry is created at the beginning of each semester to ensure proper tracking of faculty workloads and course schedules.

## 1.5 Implementation

#### 1. Manage Faculty Workload

The user can manage faculty workload by performing CRUD operations. Each faculty member will have a unique ID, name, designation, and assigned workload.

• **Add**

To add a faculty workload, the user will fill in the faculty ID, course ID, and workload hours. The following SQL query will be executed to insert the data into the FacultyWorkload table.

**SQL Query to insert data into Student table**

public void insertFacultyWorkloadFunction(int facultyID, int courseID, int workloadHours)

{

var con = Configuration.getInstance().getConnection();

SqlCommand cmd = new SqlCommand("INSERT INTO FacultyWorkload (FacultyID, CourseID, WorkloadHours) VALUES (@FacultyID, @CourseID, @WorkloadHours)", con);

cmd.Parameters.AddWithValue("@FacultyID", facultyID);

cmd.Parameters.AddWithValue("@CourseID", courseID);

cmd.Parameters.AddWithValue("@WorkloadHours", workloadHours);

cmd.ExecuteNonQuery();

}The above function will execute the query to get the maximum id of the person to insert the registration number of a student against the same id as used in the person table. Then the registration number will be stored. In this way the complete information of a student will be saved in the database

• **Update**

To update a faculty workload, the user will provide the faculty ID and the new workload details. The following SQL query will update the faculty workload.number then the id and the information to be updated is passed to a function to update it.

**SQL Query to update data Faculty Overload**

public void updateFacultyWorkloadFunction(int facultyID, int courseID, int workloadHours)

{

var con = Configuration.getInstance().getConnection();

SqlCommand cmd = new SqlCommand("UPDATE FacultyWorkload SET WorkloadHours=@WorkloadHours WHERE FacultyID=@FacultyID AND CourseID=@CourseID", con);

cmd.Parameters.AddWithValue("@FacultyID", facultyID);

cmd.Parameters.AddWithValue("@CourseID", courseID);

cmd.Parameters.AddWithValue("@WorkloadHours", workloadHours);

cmd.ExecuteNonQuery();

}

• **Delete**

To delete a faculty workload, the user will provide the faculty ID and course ID. The following SQL query will delete the faculty workload from the FacultyWorkload table.

. **SQL Query to Delete data of FacultyWorkload**

public void deleteFacultyWorkloadFunction(int facultyID, int courseID)

{

var con = Configuration.getInstance().getConnection();

SqlCommand cmd = new SqlCommand("DELETE FROM FacultyWorkload WHERE FacultyID=@FacultyID AND CourseID=@CourseID", con);

cmd.Parameters.AddWithValue("@FacultyID", facultyID);

cmd.Parameters.AddWithValue("@CourseID", courseID);

cmd.ExecuteNonQuery();

}•

**Show Facult WorkLoad**

To view all faculty workloads, the following SQL query will retrieve the data and display it in a DataGridView.

**SQL Query to Show details of Student**

public void showFacultyWorkloadFunction(DataGridView dataGridView)

{

var con = Configuration.getInstance().getConnection();

string query = "SELECT fw.FacultyID, f.Name AS FacultyName, c.CourseName, fw.WorkloadHours FROM FacultyWorkload fw INNER JOIN Faculty f ON fw.FacultyID = f.FacultyID INNER JOIN Courses c ON fw.CourseID = c.CourseID";

SqlCommand cmd = new SqlCommand(query, con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataTable dt = new DataTable();

da.Fill(dt);

dataGridView.DataSource = dt;

}

#### 2. Manage Resource Allocation

The user can manage resource allocation, including classrooms, lab spaces, and consumables

• **Add**

To allocate a resource, the user will provide the resource type, faculty ID, and allocation details. The following SQL query will insert the data into the ResourceAllocation table.

**SQL Query to insert the Advisor details**

public void insertResourceAllocationFunction(string resourceType, int facultyID, DateTime allocationDate, int duration)

{

var con = Configuration.getInstance().getConnection();

SqlCommand cmd = new SqlCommand("INSERT INTO ResourceAllocation (ResourceType, FacultyID, AllocationDate, Duration) VALUES (@ResourceType, @FacultyID, @AllocationDate, @Duration)", con);

cmd.Parameters.AddWithValue("@ResourceType", resourceType);

cmd.Parameters.AddWithValue("@FacultyID", facultyID);

cmd.Parameters.AddWithValue("@AllocationDate", allocationDate);

cmd.Parameters.AddWithValue("@Duration", duration);

cmd.ExecuteNonQuery();

}

• **Update**

To update a resource allocation, the user will provide the allocation ID and the new details. The following SQL query will update the resource allocation.

**SQL Query to Update Advisor Details**

public void updateResourceAllocationFunction(int allocationID, DateTime allocationDate, int duration)

{

var con = Configuration.getInstance().getConnection();

SqlCommand cmd = new SqlCommand("UPDATE ResourceAllocation SET AllocationDate=@AllocationDate, Duration=@Duration WHERE AllocationID=@AllocationID", con);

cmd.Parameters.AddWithValue("@AllocationID", allocationID);

cmd.Parameters.AddWithValue("@AllocationDate", allocationDate);

cmd.Parameters.AddWithValue("@Duration", duration);

cmd.ExecuteNonQuery();

}

• **Delete**

To delete a resource allocation, the user will provide the allocation ID. The following SQL query will delete the resource allocation from the ResourceAllocation table.

**SQL Query to Delete Advisor details**

public void deleteResourceAllocationFunction(int allocationID)

{

var con = Configuration.getInstance().getConnection();

SqlCommand cmd = new SqlCommand("DELETE FROM ResourceAllocation WHERE AllocationID=@AllocationID", con);

cmd.Parameters.AddWithValue("@AllocationID", allocationID);

cmd.ExecuteNonQuery();

}

• **Show**

To view all resource allocations, the following SQL query will retrieve the data and display it in a DataGridView.

**Advisor details**

public void showResourceAllocationFunction(DataGridView dataGridView)

{

var con = Configuration.getInstance().getConnection();

string query = "SELECT ra.AllocationID, ra.ResourceType, f.Name AS FacultyName, ra.AllocationDate, ra.Duration FROM ResourceAllocation ra INNER JOIN Faculty f ON ra.FacultyID = f.FacultyID";

SqlCommand cmd = new SqlCommand(query, con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataTable dt = new DataTable();

da.Fill(dt);

dataGridView.DataSource = dt;

}

#### 3. Manage Projects

The user can manage consumables, including adding, updating, and deleting consumable resources

• **Add**

To add a consumable, the user will provide the consumable name and quantity. The following SQL query will insert the data into the Consumables table.

**SQL Query to insert the required**

public void insertProjectFunction(string description, string title) { var con = Configuration.getInstance().getConnection(); SqlCommand cmd = new SqlCommand("Insert into Project values (@Description, @Title)", con);

cmd.Parameters.AddWithValue("@Description", description); cmd.Parameters.AddWithValue("@Title", title); cmd.ExecuteNonQuery();

} • **Updates**

To update a consumable, the user will provide the consumable ID and the new quantity. The following SQL query will update the consumable details.

**SQL Query to Update Consumable details**

public void updateConsumableFunction(int consumableID, int quantity)

{

var con = Configuration.getInstance().getConnection();

SqlCommand cmd = new SqlCommand("UPDATE Consumables SET Quantity=@Quantity WHERE ConsumableID=@ConsumableID", con);

cmd.Parameters.AddWithValue("@ConsumableID", consumableID);

cmd.Parameters.AddWithValue("@Quantity", quantity);

cmd.ExecuteNonQuery();

}

• **Delete Consumables**

To delete a consumable, the user will provide the consumable ID. The following SQL query will delete the consumable from the Consumables table.

. **SQL Query to Delete Project detail**

public void deleteConsumableFunction(int consumableID)

{

var con = Configuration.getInstance().getConnection();

SqlCommand cmd = new SqlCommand("DELETE FROM Consumables WHERE ConsumableID=@ConsumableID", con);

cmd.Parameters.AddWithValue("@ConsumableID", consumableID);

cmd.ExecuteNonQuery();

}

• **Show Consumable**

To view all consumables, the following SQL query will retrieve the data and display it in a DataGridView

.**SQL Query to Show Consumables details**

#### public void showConsumablesFunction(DataGridView dataGridView)

#### {

#### var con = Configuration.getInstance().getConnection();

#### string query = "SELECT ConsumableID, ConsumableName, Quantity FROM Consumables";

#### SqlCommand cmd = new SqlCommand(query, con);

#### SqlDataAdapter da = new SqlDataAdapter(cmd);

#### DataTable dt = new DataTable();

#### da.Fill(dt);

#### dataGridView.DataSource = dt;

## 1.6 Project Details

#### 1.6.1 Languages Used

* **C sharp**

The front of the desktop application is made using C sharp language

* **T SQL**

To store the data the T SQL database is used to save the records