**Student Details**

| Repository Name | DBS25P127 |
| --- | --- |
| Registration Number | 2024-CS-127 |

|  |  |
| --- | --- |
| ***Project*** |  |
| Project Title | Faculty Workload and Resource Allocation System |
| Executive Summary | The **Faculty Workload and Resource Allocation System** is a desktop-based application developed using **C# Windows Forms** with a **MySQL** backend. The primary purpose of this system is to efficiently manage the academic workload and available resources of faculty members in an educational institution. This project gave me a real understanding of how actual systems are designed, especially in managing faculty responsibilities, time, and physical resources like classrooms.  Through this project, I learned how to structure a system with multiple **user roles** such as **Admin**, **Faculty**, and **Department Head**. Each role has different levels of access and functionality, managed through a secure **login system**. Admins can manage faculty profiles, assign courses, allocate rooms, and schedule classes, while faculty members can view their workloads, schedules, and make resource requests.  One of the most interesting parts of the system was how it handles **workload assignment**. Faculty can be assigned to courses, final year projects, and other academic responsibilities. Each assignment is tracked along with hours, allowing the system to monitor whether any faculty member is over- or under-loaded. I also built a **schedule management** system that ensures there are no conflicts in room availability or faculty time using conflict-check logic before inserting schedule records.  Another important feature I implemented is **room and resource allocation**. Each classroom is assigned based on available capacity and time slots. I also added a table to keep track of **reserved hours per room per faculty**, which gave me insight into how resource allocation is planned in academic setups.  Overall, this project helped me understand how different parts of a system—users, courses, schedules, rooms—are connected. It taught me database design, C# form handling, and how to think through real-world scenarios logically and efficiently. |

|  |  |
| --- | --- |
| ***Interface Name*** |  |
| *Picture*   |  |  |  |  | | --- | --- | --- | --- | | **UI Component Name** | **Type of UI component** | **Class Attribute name,**  **Classname, attribute name** | **Linked to which database column/table** | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |   In case of column, write column name and in case of multiple columns, write query in last column | |

|  |  |
| --- | --- |
| ***Interface Name*** |  |
| *Picture*   |  |  |  |  | | --- | --- | --- | --- | | **UI Component Name** | **Type of UI component** | **Class Attribute name,**  **Classname, attribute name** | **Linked to which database column/table** | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |   In case of column, write column name and in case of multiple columns, write query in last column | |

|  |  |
| --- | --- |
| ***Business Report-1*** |  |
| *Business Report Name:* | |
| *Sample of report:* | |
| *Query to build Report:* | |

|  |  |
| --- | --- |
| ***Business Report-2*** |  |
| *Business Report Name:* | |
| *Sample of report:* | |
| *Query to build Report:* | |