**ASSIGNMENT COVER SHEET**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Course: BSc Computing (SE)** | | | | **Year: 2** | | CSY2027 | | |
| **Group Project** | | **Title: The Design and Development of a Course Management Software System** | | | | | | |
| Date due out: | Date due in: | | Extension date: | | | | | Extension agreed by: |
| **Student Names (List each member of the group)**  **Bipashu Thakuri (20416218)**  **Rabika Pradhananga (20416247)**  **Soman Maharjan (20416264)**  **Sujayan Raj Manandhar (20416271)** | | | | | | | **Tutor:** | |
| Student comment, specific request for feedback etc. | | | | | Marker’s General View of the work | | | |

**ASSESSMENT FEEDBACK:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **RATING SCALE** | **Excellent** | | **Good** | | **Satisfactory** | **Needs some more work** | | **Needs much more work** |
| Elicitation Plan/ Interview(s)/Findings (10%) |  | |  | |  |  | |  |
| Requirement Specifications Documentation  (10%) |  | |  | |  |  | |  |
| System Design Documentation  (25%) |  | |  | |  |  | |  |
| Prototype Functionality and Quality of Application Code (35%) |  | |  | |  |  | |  |
| Test/Evaluation Strategy  (10%) |  | |  | |  |  | |  |
| Group Cohesion, Teamwork and Project Management/ System Presentation (10%) |  | |  | |  |  | |  |
| Specific aspects of your assignment that the marker likes: | | | | Specific aspects of your assignment that need more work: | | | | |
| Tutor’s Signature: | | Date: | | | | | Grade: | |

**Table of Contents:**

[1. **Introduction:** 4](#_Toc74155379)

[1.1 Project Background: 4](#_Toc74155380)

[1.2 Project Aims and Objectives: 4](#_Toc74155381)

[1.3 Project Development Methodology: 5](#_Toc74155382)

[**2. Requirements Engineering:** 7](#_Toc74155383)

[2.1 Elicitation Activities: 7](#_Toc74155384)

[2.1.1 Interview plans: 7](#_Toc74155385)

[2.1.2 Interview findings: 10](#_Toc74155386)

[2.1.3 Other problem domain research: 17](#_Toc74155387)

[2.2 Requirements Specification: 25](#_Toc74155388)

[2.2.1 Problem Domain Description: 25](#_Toc74155389)

[2.2.2 Functional Requirements: 28](#_Toc74155390)

[2.2.3 Performance Requirements: 38](#_Toc74155391)

[2.2.4 Design Constraints: 39](#_Toc74155392)

[2.2.5 Commercial Constraints: 40](#_Toc74155393)

[2.2.6 Acceptance Tests: 42](#_Toc74155394)

[**3. System Analysis and Design:** 42](#_Toc74155395)

[3.1 Preliminary Design Stages: 43](#_Toc74155396)

[3.1.1 Textual Analysis: 43](#_Toc74155397)

[3.1.2 Significant Event Analysis: 44](#_Toc74155398)

[3.2 Detailed Statistic System Designs: 54](#_Toc74155399)

[3.2.1 BON Class Charts: 54](#_Toc74155400)

[3.3 Other Diagrams: 60](#_Toc74155401)

[3.3.1 Data Flow Diagrams: 60](#_Toc74155402)

[3.3.2 Class Diagram: 70](#_Toc74155403)

[3.4 System Database Design: 71](#_Toc74155404)

[3.4.1 E-R Model: 71](#_Toc74155405)

[3.4.2 Attribute Listings: 73](#_Toc74155406)

[**4 System Interface Designs:** 80](#_Toc74155407)

[4.1 Draft Interface Designs: 80](#_Toc74155408)

[4.1.1 Wireframes: 80](#_Toc74155409)

[4.1.2 System Navigation Diagram: 82](#_Toc74155410)

[4.1.3 System Screen Mockups: 86](#_Toc74155411)

[4.1.4 System Activity Event Diagrams: 98](#_Toc74155412)

[4.2 Design Revisions: 106](#_Toc74155413)

[4.3 Heuristic Evaluation: 106](#_Toc74155414)

[**5 System Build and Technical Notes:** 108](#_Toc74155415)

[5.1 Technically Difficult Code Sections: 108](#_Toc74155416)

[5.2 On-going Testing Methodology: 113](#_Toc74155417)

[5.3 Final System Interface Displays: 114](#_Toc74155418)

[**6 Test Strategy:** 120](#_Toc74155419)

[6.1 Overview of Test Strategy: 120](#_Toc74155420)

[6.2 Sample Test Results: 120](#_Toc74155421)

[**7 System Evaluation:** 129](#_Toc74155422)

[7.1 Usability Evaluation Strategy: 129](#_Toc74155423)

[7.2 System Plot Trials: 130](#_Toc74155424)

[7.3 System Trial Results: 139](#_Toc74155425)

[**8 Project Conclusions:** 140](#_Toc74155426)

[**9 Project Management:** 141](#_Toc74155427)

[9.1 Project Gantt Chart: 141](#_Toc74155428)

[9.2 Project Meeting Minutes: 144](#_Toc74155429)

[9.3 Project Quality Plan / Strategy: 147](#_Toc74155430)

[9.4 Project Work Log: 148](#_Toc74155431)

# Introduction:

With quality education getting prioritized, it is common for colleges to have an online course management to simplify the learning experience for their students.

Woodlands University College (WUC) is a Higher Education Institution that offers a variety of courses. Dr. Simon, the course leader of the computing department thinks that it is fit for the existing system of the university to be automated. To start off this process, only the computing course is to be digitized, which is likely to be followed up by other available courses.

The prototype project is being developed keeping in mind utmost ease and convenience of the student body, and both administrative and teaching staff of the university. Aspects of this project focus on improved record management, better communication between teachers and students, and dissemination of authentic information.

## 1.1 Project Background:

Dr. Simon, the course leader of the Computing department of the Woodlands University College (WUC), requested the digitization of the Computing degree course. The digitization process includes improved services for both the students and the staff. These facilities consist of storing and updating student, staff, and course records, regulating modules, attendance, as well as assignments, generating reports, organizing diaries, and assigning personal tutors and timetables.

## 1.2 Project Aims and Objectives:

The main objective of this project is to transform the course management of the university’s Computing department from a clerical to a computerized system so that authentic data can be accessed conveniently. The new system also aims to overcome the limitations of a paper-based system which include data redundancy, data loss, and poor management. Different functionalities are to be added keeping in mind that the site is user-friendly, cost-effective, and reliable. Furthermore, the system is to be flexible for future modifications, as it may be implemented for all the courses offered by the university.

The goal is to design and implement a pilot system with key features given below:

* To store, display, edit, delete and archive student and staff records
* Enroll students into the courses they choose and display necessary information about the courses
* Assign students to modules based on the respective course that they have chosen
* Create and display new/submitted assignments and display grades and feedback as provided
* Record attendance of both students, search attendance records of a particular student, take action if attendance is poor
* Assign personal tutors to students
* To make and delete diary entries
* To create, display, and print reports

## 1.3 Project Development Methodology:

Implementation and Testing

System Interface Design

System Analysis and Design

Requirement Engineering

Maintenance

1. **Requirement Engineering:**

Requirement Engineering is the process of finding, defining, and documenting the requirements of a project. We did this by interviewing our stakeholders: Dr. Simon White, Mr. Adam Blake**,** Dr. Raj Singh, and Mr. Mark Williams.

The process is given below:

* Understanding the current model of course management and identifying the problem domain by interviewing the stakeholders.
* Requirement analysis is done by separating the essential needs of the client in order to avoid future conflict
* With the collected data, functional requirements such as university rules and needs of the system are identified.
* Performance specification and commercial constraints are identified and aligned with other necessities.
* All functional, non-functional, and technical requirements are documented and validated via stakeholders.

1. **System Analysis and Design:**

System analysis is the process of studying requirements and problem domain to identify its purpose and ensure proper procedure and system. It is done in order to:

* identify the objectives of our system
* interpret elements of a system to implement in final system design
* develop a highly efficient and effective final project
* SRS document is made here.

1. **System Interface Design:**

System interface design is the process of designing user interfaces for our targeted project. It mainly focuses on the overall view; how the user perceives it in their respective devices. The goal is to always make it fun and satisfying regardless of the platform. This can be achieved using wireframes, where the front-end design and designing BON charts for system interface design can be pre-planned.

1. **Implementation and Testing:**

The implementation is done once the design phase is completed. This is the phase where the computer programs are created. The actual product is now developed with coding and high-level language is used to build the system. The logical data collected till now converted into physical form using algorithms and charts. The SRS document is used as a reference here as well. After implementation, testing is carried out where the program is executed with the intention of finding errors. After the bugs are identified, debugging is done to analyze and fathom the bugs in order to make the program work as expected. For our scenario, we are going to use HTML, CSS, JavaScript, Laravel, and PHP for the application of the code.

* Implementation is done to test the functionality and practical feasibility of the system.
* Testing can be done in two way:
  + Black box testing: It is done by a person without any knowledge of coding. It only checks the behavior and functionalities of the software so that the user requirements are fulfilled. It is from an end user's perspective.
  + White box testing: It is done on the internal structure of the software i.e. coding. It is done to improve the flow of input and output through the system hence improving design and usability. It is based on the inner working of the application and revolves around internal testing.

1. **Maintenance and Support:**

It is the last phase of requirement engineering. The goal here is to fix any software bugs identified during the usage of the system. The software is updated and tested for optimal user experience.

# Requirements Engineering:

Requirements engineering is the process of finding out the problem domain and necessities of the client from which a solution is designed and documented. It is an important process as it is the key to understand the problem domain and strengthen the base for the final solution. Moreover, it helps to find the constraints that the system needs thereby making it compatible in the technical, operational, and physical (university) areas. It is also known as requirement elicitation where the goal is to produce a practical solution, taking into consideration the needs, wishes, and constraints of the client, along with the cost and deadline of the project.

## 2.1 Elicitation Activities:

To gather information, the prime sources include the client specification document itself, interviews conducted with the client and potential users, and study of similar existing systems.

### **2.1.1 Interview plans:**

The project stakeholders include students, administration, and teaching staff so interviews are to be held accordingly. A detailed interview plan with questions for each individual and their role is given below:

|  |  |
| --- | --- |
| **Interview with Dr Simon White (The computing course leader)** | |
| **Question** | **Question** |
| **Number** |  |
| 01 | Who will the potential target users of this software in your opinion? |
| 02 | In what ways will the potential future users of the system benefit? |
| 03 | What should the report look like? (what information should be in the report) |
| 04 | What is Diary Management used for and what does it mean by “Initiate Automated Action”? |
| 05 | What is your relevant business model coping to adjust with this software? |
| 06 | Could you provide some outline on your design constraints? |

|  |  |
| --- | --- |
| **Interview with Mr. Adam Blake (A course administrator)** | |
| **Question** | **Question** |
| **Number** |  |
| 01 | How would you describe the functionality of the system? |
| 02 | Could you outline some features that you would like to see in the new system? |
| 03 | Who should have access to Course Management System? (Staff, Student, admin) |
| 04 | Should the Assignment Management part of the system have functionality for student to submit their assignment? Or is it just for staff to assign assignments? |
| 05 | How will you tackle future software problems and probable ransom ware attacks? |
| 06 | Could you specify the cost and deadline of the project? |

|  |  |
| --- | --- |
| **Interview with Dr. Raj Singh (A senior lecturer, module leader and personal tutor)** | |
| **Question** | **Question** |
| **Number** |  |
| 01 | How do you currently address your students' queries and how would you change it? |
| 02 | From what means do you currently distribute module materials and how would you prefer to distribute it now? |
| 03 | What improvements would you like to add in order to enhance the lecture experience for the students? |
| 04 | Is there any specific feature you would like to add? |
| 05 | Is there any feature you think that the current system possesses, that should be implemented to the new system? |
| 06 | How do you conduct pat sessions and how would you like to utilize this site to improve on it? |

|  |  |
| --- | --- |
| **Interview with Mr. Mark Williams (an existing student)** | |
| **Question** | **Question** |
| **Number** |  |
| 01 | How do you get notified about new assignments currently? Is the information flow through this process efficient? |
| 02 | Do you monitor your attendance for different modules yourself? If not, does a warning reach you when your attendance is almost below minimum or after it is below the minimum required percentage? |
| 03 | What major difficulties are you facing because of the current course management system? |
| 04 | Are there any aspects of the current model that you think are very efficient? |
| 05 | What are some features that you want to suggest for the new system? |
| 06 | Have you used any other course management systems previously? If so, could you briefly share your experience highlighting your likes and dislikes about it. |
| 07 | What kind of training would you want to be conducted for the students so that they can benefit as much as possible from the site? |

### **2.1.2 Interview findings:**

The findings after holding the interviews successfully with all the targeted individuals are given below:

**Interview with Dr. Simon White (The computing course leader)**

Interview Date: 22nd December, 2020

Duration: 20 minutes

Interview attendees:

* Bipashu
* Rabika
* Soman
* Sujayan

All questions and respective responses of this interview are shown in the following table:

|  |  |  |
| --- | --- | --- |
| Topic Area | **Motivation and Client Constraints** | |
|  |  |  |
| **Interviewer** | **Question** | **Question - Client Response** |
|  | **Number** |  |
| Rabika | 01 | Who will the potential target users of this software in your opinion?   * Student, module leader, tutors |
| Rabika | 02 | In what ways will the potential future users of the system benefit?   * Students are one of the end users of the system who will especially benefit from this change as they can track and submit their assignments easily as well as receive feedback and keep track of their grades, they can also keep an eye on their attendance through notifications. Easier communication is established between the three bodies that is administration, tutors and the students. Teachers can share study materials through the platform and monitoring progress of the students and providing personalized help becomes easier. The administration can also easily keep track of all the students and staff in the university without problems such as data loss or redundancies. |
| Rabika | 03 | What should the report look like? (what information should be included in the report)   * Sample reports that are being used currently by the university will be provided for reference and the current layout should be followed in the new system. |
| Rabika | 04 | What is Diary Management used for and what does it mean by “Initiate Automated Action”?   * The scope for diary management is yet to be discussed and finalized. |
| Rabika | 05 | What is your relevant business model coping to adjust with this software?   * Currently spreadsheets are being used to keep track of all the information regarding the staff and students of the university including the courses being offered, contact details of everyone or the grades that the students have received every term. All these details would need to be stored in the new system. Unlike the previous system, updating records in one place should be enough to store the updated information and for it to be accessed by the targeted individuals. |
| Rabika | 06 | Could you provide some outline on your design constraints?   * There are no specific color themes or fixed design for the site that is to be strictly followed. Suggested themes for reference include Blackboard (has a dark theme with text in white). |

**Interview with Mr. Adam Blake (A course administrator)**

|  |  |  |
| --- | --- | --- |
| Topic Area | **Motivation and Client Constraints** | |
|  |  |  |
| **Interviewer** | **Question** | **Question - Client Response** |
|  | **Number** |  |
| Rabika | 01 | How would you describe the functionality of the system?   * The system is not centralized and it is difficult to track all the subjects being offered by the university including details about them like credit hours or the deadline of assignments. All information is being maintained using a spreadsheet by the administration and if any staff wants to access the information then we send a formal email. This method makes it difficult to maintain data integrity as this data needs to be updated regularly and to access the latest list, the whole process of sending and receiving emails needs to be repeated. |
| Rabika | 02 | Could you outline some features that you would like to see in the new system?   * As mentioned before about the difficulties of the current system regarding the storing and distribution of data, it would benefit the university if all the courses offered by the university and details regarding them were displayed in one page. Also the staff should be able to update the information and the latest version of this list should be accessed by the staff as well as only necessary information should be accessible by the students. Distributing study materials effectively should also be one of the aims of this system. |
| Rabika | 03 | Who should have access to Course Management System? (Staff, Student, admin)   * These three user groups are the targeted user groups of the system and including these in the system will be sufficient. |
| Rabika | 04 | Should the Assignment Management part of the system have functionality for student to submit their assignment? Or is it just for staff to assign assignments?   * This system should not only benefit the staff but also the students of the university so new functionalities should be added keeping in mind that they are advantageous to the students as well. So tutors should be able to create assignments which the students enrolled in the course should receive and be able to submit them as well.   The tutors should also have the authority to select whether the assignments can be submitted multiple times, and after students submit their work they should have access to the files and then publish grades. |
| Rabika | 05 | How will you tackle future software problems and probable ransom ware attacks?   * Use of hardware devices like firewall to keep track of incoming malware or attacks on the system. In addition, regular penetration testing will be held to detect any vulnerabilities and to check if anything is malfunctioning. All end users will be educated on different kinds of viruses and ransom-ware attacks, for example, on topics such as not opening suspicious emails or downloading attachments by unknown senders and immediately informing the IT department if such happens. The risks cannot be minimized fully but necessary measures will be taken to prevent issues on the system. |
| Rabika | 06 | Could you specify the cost and deadline of the project?   * According to all the requirements provided, a cost should be quoted and further discussion is to be held on the matter of the budget. |

Interview Date: 22nd December, 2020

Duration: 10 minutes

Interview attendees:

* Bipashu
* Rabika
* Soman
* Sujayan

All questions and respective responses of this interview are shown in the following table:

**Interview with Dr. Raj Singh (A senior lecturer, module leader and personal tutor)** Interview Date: 22nd December, 2020

Duration: 10 minutes

Interview attendees:

* Bipashu
* Rabika
* Soman
* Sujayan

All questions and respective responses of this interview are shown in the following table:

|  |  |  |
| --- | --- | --- |
| Topic Area | **Motivation and Client Constraints** | |
|  |  |  |
| **Interviewer** | **Question** | **Question - Client Response** |
|  | **Number** |  |
| Sujayan | 01 | How do you currently address your students' queries and how would you change it?   * I repeatedly check my email with a few day gaps and reply to their questions but sometimes I miss their emails and their queries get overlooked. This usually happens when the student does not reply in the same email chain therefore the conversation is easily lost. I would like to suggest a chat feature so that this problem can be avoided in the new system. |
| Sujayan | 02 | From what means do you currently distribute module materials and how would you prefer to distribute it now?   * I mostly share the slides via email and sometimes via Google drive. This method makes it difficult to track progress such as to find specific slides of the latest week. I would prefer a system which allows me to upload a link (URL) through which all the students can easily view the slides. |
| Sujayan | 03 | What improvements would you like to add in order to enhance the lecture experience for the students?  I would like to suggest a folder mechanism through which assignments can be easily viewed and managed, it should also be accompanied by access rights privilege, and where the grant to edit or access personal files should be allowed to authorized personals only. The module leader should also have the right to revoke access after the submission date is past due. The site should also consist of the class recordings which the students can search via keywords as well. |
| Sujayan | 04 | Is there any specific feature you would like to add?   * I personally would like a file compression system as downloading and viewing the files for assignments takes up a lot of time. |
| Sujayan | 05 | Is there any feature you think that the current system possesses, that should be implemented to the new system?   * Like in the previous system, I would like a similar role management system as well as strong security and data backup system. |
| Sujayan | 06 | How do you conduct pat sessions and how would you like to utilize this site to improve on it?   * I would like an online conversation platform such as links to Google meets so I can conduct video conferences, also if I am not online, then a telephone service could be provided to directly contact me. |

**Interview with Mr. Mark Williams (an existing student)**

Interview Date: 22nd December, 2020

Duration: 20 minutes

Interview attendees:

* Bipashu
* Rabika
* Soman
* Sujayan

All questions and respective responses of this interview are shown in the following table:

|  |  |  |
| --- | --- | --- |
| Topic Area | **Motivation and Client Constraints** | |
|  |  |  |
| **Interviewer** | **Question** | **Question - Client Response** |
|  | **Number** |  |
| Sujayan | 01 | How do you get notified about new assignments currently? Is the information flow through this process efficient?   * Notifications about new assignments reach me through emails which is difficult for me to track as I have to filter through all my emails to access my assignments. I also face issues keeping track of how I have to submit my assignments, for example submitting a zip file or simply pasting the code in a document. In addition, after submitting assignments we do not receive constructive feedback or a consolidated report of our work as sending individual emails to all the students must be a hassle to the teachers as well |
| Sujayan | 02 | Do you monitor your attendance for different modules yourself? If not, does a warning reach you when your attendance is almost below minimum or after it is below the minimum required percentage?   * The respective module leader takes our attendance for each module. We have to personally mail them to check if our attendance is enough or not. After sending the mail we have to wait for 2 days or more, and if not, we are informed before taking the examination. |
| Sujayan | 03 |  |
|  |  | What major difficulties are you facing because of the current course management system?   * In this email-based system, to view an assignment, we need search for the respective email chain of the module leader and subject. If we desire to view slides or previous reference materials, we need to search through the entire database consisting all emails received which is a huge hassle. |
| Sujayan | 04 | Are there any aspects of the current model that you think are very efficient?   * The current system is managed firsthand by the administration, the data is accessed through a physical database hence unauthorized people are not allowed to view information easily. |
| Sujayan | 05 | What are some features that you want to suggest for the new system?   * I believe an automatic checking system would be the feature I would like the most in the new system as the current system fails to deliver us our grades on time. Also, a notification system to make us aware about pending assignment would be helpful. |
| Sujayan | 06 | Have you used any other course management systems previously? If so, could you briefly share your experience highlighting your likes and dislikes about it.   * Yes, I have, the system I know of consisted of slides being uploaded and accessed via Google classrooms, and attendances being managed via excel spreadsheet. I would hope to see a security check before accessing information as well. Also, a similar role management system where only module leaders are allowed access to student attendance would be optimal. |

### **2.1.3 Other problem domain research:**

#### 2.1.3.1 Comparable Software System Review:

Comparable Software System review is one of the most important process in requirement engineering for domain research activities. It is done by studying a similar existing system and analysing it to know more about the system and its constraints, ideas and backgrounds. It also assists engineers in validating the required quality, functionality, components and other vital features of the software. This helps to decide whether a particular type of software is compatible or not. To ensure this, similar websites were thoroughly examined and are mentioned below:

* Moodle

Moodle is an open source, user-friendly learning management system being used all around the globe by many institutions, universities, and organizations. It is the most used learning system in the world.

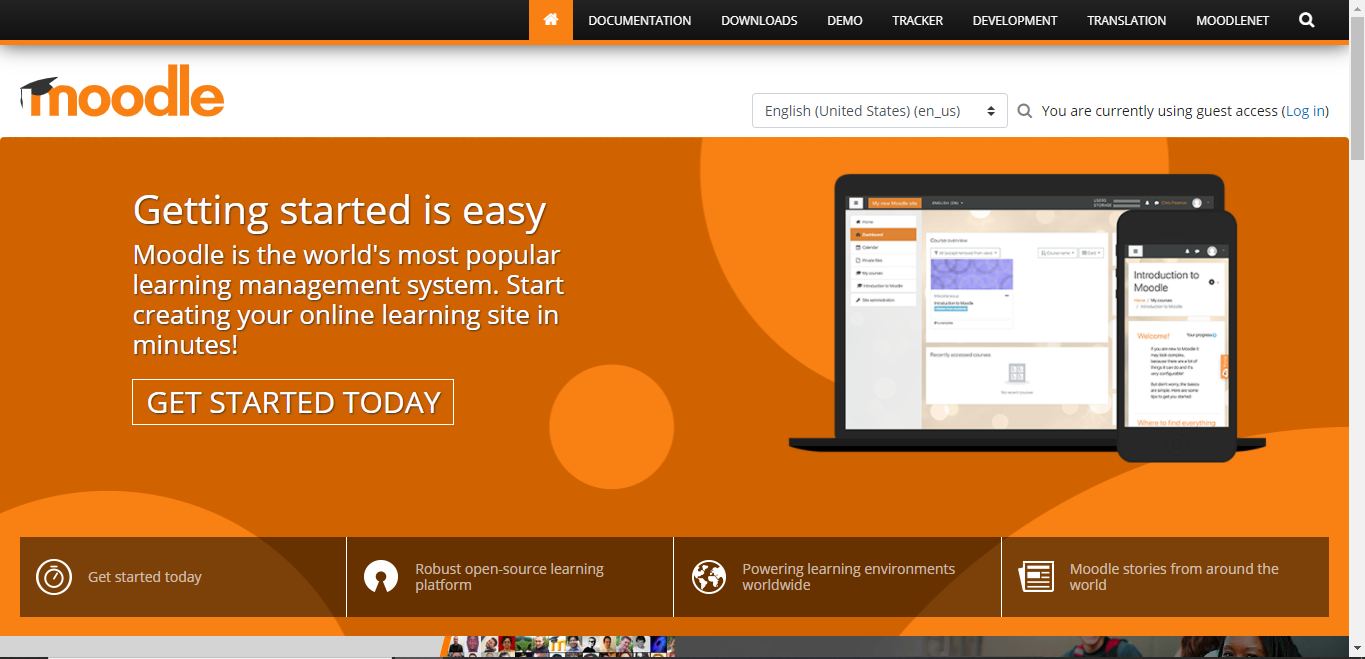


Fig: Moodle Interface

Some of the positive and negatives aspects of this system is discussed below:

|  |  |
| --- | --- |
| Positive Aspects | Negative Aspects |
| * It is based on open-source technology meaning it is freely available to anybody without fees and security bugs can be identified faster due to visible code. * It is easy to customize. * It is user-friendly in a sense that it has a simple interface, easy navigations and the design is responsive meaning we can access it through mobile phones too which makes it so much convenient. * Student records can be added, updated and deleted by admins, tutors and course leaders when necessary. * The materials uploaded by tutors and course leaders can be easily accessed by each students. * Grades are visible. | * It is not fully developed to subsist with consequential projects. * Since it is used by a large number of students, buffering may occur while giving tests and examinations online. * May need some knowledge on coding and programming to customize. * It is a slower system. |

* Nile

Nile is a website used by the Northampton University in London as its course management system. It is the closest one in terms of functionalities required by our stakeholders.

Some of the positive and negative aspects of Nile are as follows:

|  |  |
| --- | --- |
| Positive Aspects | Negative Aspects |
| * Tutors can assign grades with messages to each individual student. * All of the functionalities are presented with neat and clean look on the opening page. * Grades are private in sense each students can view only their information, not of others. So, it is more reliable and secure. * Grades are also specific to each subject. Grades-display is subject and student wise. * Announcements are made with each new amendments. * Students can customize their own information. * Adding of course and module activities is controlled by admin and course leader. * Students-log-in and Tutors- log-in are separate which offers safe communication of information between both parties. * It is media-friendly software system. | * Frequent buffer. * Too congested, hence may be complex to use. * It does not have an easy navigation system like Moodle. |

* Blackboard

Blackboard is an online based course management, learning, teaching and knowledge communication system. It grants students to indulge in online classes and use materials for a particular course. It also compliments face-to-face teaching and learning with people around the globe.

Just like Moodle and Nile, Blackboard also has its positive and negative aspects which are mentioned below:

|  |  |
| --- | --- |
| Positive Aspects | Negative Aspects |
| * It has wide range of functionalities. It enables instructors to supply students with various features such as provision of course materials and assignments, online chats, online quizzes, discussion boards etc. * The communication between students and instructors is easy. * Grades are provided in an electronic format to the students. * Instructors can put feedbacks on quizzes and assignments. Students can access those information immediately. * It is easy to make amendments such as adding, updating deleting academic records such as student records, module activities, course contents etc. * No technical knowledge is required in order to use Blackboard. * It is also media friendly. It can be used in any type of device such as smartphones and tablets * It provides full features for creating courses, providing live training and lectures and management of users. | * Interface is obsolete. * Slow loading of pages and occurrence of buffers. * Cannot be customized as per user requirements. |

* Canvas

Canvas is an online course management system like Blackboard. Many colleges and universities across the globe use this. Canvas is certainly better designed and is more advanced than Blackboard. However, it also has its fair share of positive and negative aspects which are mentioned below:

|  |  |
| --- | --- |
| Positive Aspects | Negative Aspects |
| * It provides full features for creating courses, live trainings and managing users. * It permits professors and module leaders to post information online such as grades and assignments. * It also offers chat rooms so that the communication between instructing and learning party is effective and regular. * It can be customized as per user requirements. * It allows educators to manage course tools such as adding new course materials, deleting and editing. * User Interface is very intuitive and even non tech authorities can use it effectively. * It is media friendly. * It allows learners to share documents. * It allows users to assign teachers, assistants and students as well. * It has functionalities for sending messages, making comments and assigning works. * Notifications about announcements and activities are made. * Moreover, the layout of the website is visually pleasing. | * False submission of assignments are visible. * Sometimes notifications about new announcements may not reach the user. |

#### 2.1.3.2 Development Relevant Legislation:

Development Relevant Legislation is a set of protocols passed by the government to be adopted in the development of a system project. They are the by-laws proposed by relevant government officials. Some of these are as follows:

* The UK Data Protection Act (DPA):

Data Protection Act fully incorporates all the requirements of EU general data protection regulation. It ensures that everyone’s data is treated safely and is only used for its intended purpose. It brings with it increased accountability. Failure to apply this could bring implications such as large financial fines or personal prosecution of individuals.

Mother provincial has overall responsibility for data protection and she is supported by data protection officer. They will ensure policies are in place. Mother superior takes responsibility in each home supported by the Data Protection Lead which is usually the trailer. They will help the staff to understand the responsibilities and ensure their policies and procedures are implemented. Its responsibility of all staff to follow policies and local procedures to ensure that is kept safe and identify and report any data incidents in case they are the organization who support all homes in England with ITA so they will take responsibility for insurance security around any IT systems.

One of the categories of the information is personal information which includes name, address, contact details, gender, birthdate, geographic location and so on. Such information leads to identification of individuals. Another category is sensitive information which include racial or ethnic origin, political opinions, religious beliefs, genetic information. Loss or wrongful exposure of this information could potentially lead to significant harm to the individuals concerned. Therefore, data incidents concerning sensitive information would lead to increased fines and prosecution.

The 6 DPA principles are:

1. Personal data shall be processed lawfully, fairly and in transparent manner.

That means people must be informed about what information will be kept about them, how it will be stored and who it may be shared with.

1. Personal data shall be collected for specified, explicit and legitimate purposes and not further processed in a manner that is incompatible with other purposes.
2. Personal data shall be adequate, relevant and limited to what is necessary in relation to the purposes for which it is processed.
3. Personal data shall be accurate and, where necessary kept up to date.
4. Personal data should be kept in a form which permits identification of data subjected to necessary purposes.
5. Personal data shall be processed in a manner that ensures appropriate security of the personal data, including protection against unauthorised or unlawful processing and against accidental loss, destruction or damage, using appropriate technical or organisation measures.

* The Privacy Act (2020):

The Privacy Act regulates the way an individual’s personal information is handled. So, it gives every individual a greater influence over how their personal information is manipulated.

This act came into effect since December 1st, 2020; introducing a number of new protection of privacy laws and obligations for organizations, including mandatory privacy breach notifications, access directions and compliance notices.

Some of the rules of this act are:

1. Many organizations have experienced hacking attempts or cyber-attacks or have accidentally lost or disclosed customer information. And under this law, if an organization has a serious privacy breach, the Privacy Commissioner and in most cases the affected people must be informed. Every organization must comply with it.
2. The Privacy Act will not directly impact day to day work, as it applies to federal government institutions.
3. It lays out the regulations on how personal information is used, collected and managed.
4. Each province and territory has a similar act regulating how provincial government institutions collect, use and manage personal information.
5. If an individual has requested an information about themselves, an access direction requires an agency to provide that information to the individual. There are offences like misleading an agency to get someone’s personal information for which up to ten thousand dollars may be fined.
6. An organization should only collect information needed for lawful purposes, this includes customer numbers and other unique identifiers.
7. The information collected from young people is to be fair and reasonable, as they might not realize the consequences of sharing their information.
8. There is also principle about sending people information overseas. It’s much easier to send information overseas. So, this Act also includes rule to check what organizations should know be before sharing information overseas.

* Digital Economy Act (2017):

Digital Economy Act (2017) is an act of the parliament of the United Kingdom which makes provisions about electronic communications, infrastructures and services to give all premises in the UK a legal right to request a minimum standard of broadband connectivity, expected to be 10 megabits per second (Mbps). The key provisions of this Act are as follows:

1. In order to provide digital government, this Act allows sharing of data between government departments. Digital government is the use of ethnical communication devices, such as computers and internet, to provide public services to citizens and other persons in a country or region.
2. There is a penalty for any type of unwanted, unsolicited calls made.
3. ISP (Internet Service provider) is required to use internet to block any adult contents.
4. Creating a UK-age-verification regulator to publish guidelines about how pornographic websites which operate on a commercial basis should ensure their users are aged 18 or older.
5. Introducing a Universal Service Obligation which allows users to request broadband speeds of at least 10 Mbps.

It altogether results in greater information and choice among customers, saves time and ensures personalization of customers and also creates important data which can give new perception.

* Computer Misuse Act (1990):

Computer Misuse Act is an act of the parliament of the United Kingdom government. It was introduced in 1990. There are four main offences involved in this Act. It came into place to reduce cyber-crime like hacking. They are as follows:

1. Unauthorised access to computer material:

It is where you might know someone’s password and just look at their files. The charges are six months prison and larger fine.

1. Unauthorised access with intent to commit or facilitate a crime:

It is where someone intends to break the law after logging on as someone else. Using someone eBay account to buy something is one of the examples for it. The charges are five years prison and larger fine.

1. Unauthorised modification of computer material:

It includes deletion of file, changing files, introduction of viruses once you are logged on as someone else.

1. Making, supplying or obtaining material that could be used in computer misuse offences:

It includes creation of viruses, giving other people viruses to use, getting viruses off other people to use.

This Act ensures that the internet must be cautiously used with ethics and there must not be any cyber-crime like data stealing, hacking, online robbery etc. Otherwise charges would be taken.

#### 2.1.3.3 Any other relevant problem domain investigation data:

WUC is intending to make its system online to simplify the learning process for students and reduce problems it has been facing because of its clerical system. Such issues are discussed below:

1. The management of record was difficult. Since, this system needs to be updated daily, data is physically entered into spreadsheets. This made record management difficult due to loss of huge amount of space, and due to poor management results were corrupted.
2. Assignments and grades were not duly provided.
3. Difficulty in communication between three bodies which are administration, tutors and students.
4. Loss and repetition of data.
5. Same data being present in more than one place resulting in requiring more than one update of the same data.
6. Difficult to track minute information of the course subjects such as credit hours or the deadline of the assignment.
7. The process of sending and receiving emails to authorized staff had to be done in order to access information. Same process is required for minute updates as well.
8. Staff are not able to update their own information.
9. Ineffective distribution of course materials.
10. Difficulty in course content management.
11. Students facing difficulty in accessing assignments because of having to keep track of them through emails.
12. Maintaining attendance records is difficult since tutors have to send these records to admins.
13. Sending individual reports to students for each module is difficult for a teacher. Students have not been receiving necessary feedbacks on submitted assignment on time.
14. Due to unmanaged documents, finding a certain file requires scrolling through a lot.

## Requirements Specification:

This segment will consist of a description of the functional and non-functional requirements.

### **Problem Domain Description:**

The sector is in which issues may arise is considered to be the problem domain. Analyzing and resolving such issues will enhance the user experience and resources. This portion of the project will deal with identifying the problems and restrictions of the previously existing business.

#### Existing Business Operation:

In the existing system, the administration is responsible for manually managing the information of the students. Even though this can provide better security, it causes is a hassle for the staff and can be a lengthy and inefficient process. Some activities include:

Checking attendance:

Currently, to view whether a student has adequate attendance, he/ she must first send an email to the respective module leader. The teacher then has to communicate with the administration to get the records. After the tedious process of reviewing a physical database, the staff member can provide information to the tutor about the student. Henceforth the tutor can reply to the student’s query. Aside from the extensive process, the student cannot double-check if his/her attendance was marked or missed by a teacher immediately after class. Also, if multiple students were to enquire about their attendance, then it would overload the administration.

Diagram

Description automatically generated

Figure 2.1.1.1.: Check attendance flowchart

Receiving and following up on assignments:

The current structure only provides an email system to share course materials. After a teacher has sent course materials, many students reply to the mail or submit their assignments via the same email chain. The lack of proper management confuses the teacher and thus delays his reply. This may also result in student queries being missed by the teacher. Since teachers aren’t active in emails daily, students have to wait a considerable time to receive a response; or even feedback and grades for submitted assignments. In case of grading the assignment, the posted grade might not be confidential.

Diagram

Description automatically generated

Figure 2.1.1.1.: Receiving and following up on assignments flowchart

Updating records:

The present system is hard to maintain. The records are manually entered and only one person can access the database at a time. The module leader and administration have to communicate daily to record the attendance. Furthermore, any changes to staff or student detail can be only done by mailing to the administration. Lastly, the database is not backed up therefore data might get permanently lost in an accident.

Diagram

Description automatically generated

Figure 2.1.1.1.: Updating records flowchart

#### 2.2.1.2 Summary of existing business limitations requiring resolution:

Limitations help identify areas for improvement. The restrictions of the current system are:

• The poor management makes retrieving data difficult

• Queries and assignment submission procedure are not proper

• Reviewing mails is difficult as there is no form of separation between modules or announcements

• There is no back-up system for data as a natural disaster can cause permanent loss of data

• The PAT sessions are not recorded; therefore, students cannot review sessions

• Almost every process has to go through the administration, therefore, overworking the staff.

### **2.2.2 Functional Requirements:**

Functional requirements deal with the actual system as well as its components and the features that the system must possess. Defining the functional requirements creates a clear picture of what features the system must contain which makes development work easier. The current system of the university is a clerical one, so the new system is to be made with an aim to overcome all the limitations of it. Reducing manual work by fostering automation in daily tasks, increased security and improved time management are a few goals of the new system.

The university currently makes use of spreadsheets to store all kinds of data related to the university. The disadvantages of this include increased likelihood of human errors, lack of collaboration, chances of data loss as well as data fraud and so on. Only one person can edit data at a time and in case there are any errors made, it is likely that the previous version of it cannot be restored if daily backups are not made. Sharing data among other people is also time consuming as they would need to be sent these files as soon as they are updated which happens frequently in the system.

According to the client proposal, there are ten different key areas to work on with individual functionalities for each area. These are described in detail below:

* Student Records:

Records of all the students associated with the university are to be stored in a database of this new system. Only an administrator has the rights for creating a new student record. Information such as the student’s full name, date of birth, the course that they are taking, their email address, contact information and so on are to be entered while creating a new record. These records can later be edited, displayed or deleted. Students can then view their profile after they log in. Records can also be archived and this function can be used to store data of students who have already graduated from the university. A function to unarchive these records will also exist in order to reverse any error

* Staff Records:

Staff records consist of both teaching staff as well as administrative staff that are linked with the university. The admins have the right to add new records for the teachers. Similar to student records, their full name, contact information, date joined, salary and so on are to be stored. Teachers can also view their information after they login in their profiles which can also be edited, deleted or viewed by the admins. Data of staff which are no longer associated with the university can be stored in the archived section. These records can also be unarchived as previously mentioned.

* Course Records:

All the courses that are offered by the university and details of the courses are to be stored under course records. An admin can add, view the list and details of all the current courses in the system, modify information about the courses or delete them. Details about the course such as the course title, credit score, description of the course, number of semesters, start and end date are to be stored. The admin also adds a course leader as well as different modules that are to be taught under the course during the process of creating a new record for a course. Students can then view details of the course that they choose to take. Information about old courses can also be stored in the archived records section which can also be unarchived similar to other records.

* Module Management:

All courses contain a number of modules that fall under them meaning a student must study all these modules for the completion of the course that they are taking. Similar to adding records of new courses, only admins can create module records. Details such as the title of the module, start and end date, module code and so on are stored. A module leader is also assigned, and a module can be associated with more than one course. The list of modules that a teacher gives lessons for can be seen in their individual profiles when they log in. Teachers can also add different materials such as assignments for a module that they teach. Students can then view all the modules as well as materials posted by teachers when they log in. Modules can also be archived or unarchived similar to courses.

* Assignment Management:

Teachers can post assignments for different modules and to create a new assignment, details such as the title of the assignment, description, submission date are to be entered. Files can also be uploaded so that students can just download them and access them offline as well. A module is also to be specified before uploading an assignment.

Students can view assignments sorted according to modules in their profile, they can download attached files that contains information about a certain assignment and then post them after completion. Work that is posted by the students can be accessed by teachers associated with the module, who can then review the work of the students and provide with feedback and grades. Data about assignments that have passed the submission date can be stored using the archived feature.

* Attendance Management:

Admin and staff can create attendance records for various students. Students can either be marked present or absent for different modules that they take. These records can also be edited, deleted, archived, or unarchived. These authorities can also report a student in case their attendance records are poor. Records for a particular student can also be searched on the basis of their name or email address. On the student’s side, they can also view their attendance records for different modules.

* Personal Tutor Management:

Personal Tutors are assigned to establish a better relationship between students and tutors by improving communication between them. So, the list of all available tutors is to be displayed in the dashboard of an admin who can then assign a personal tutor to a student. Tutors can view all the students that fall under their care and information such as their student group’s email address and contact information can be seen by the tutor. Students can also view similar information about their personal tutors on their profile. This way communication can be initiated from both ways.

* Timetable Management:

Timetables can be created by the admin for different modules containing information such as module name, day, start time, end time and so on. These records can be edited, deleted, displayed, archived, or unarchived per necessity. Students can then view their timetable with an option to filter by days of the week with information such as the module name, teacher and so on. The staff can also view these timetables to get information about the timing of the classes for the modules they teach.

* Diary Management:

All the students, teaching and administrative staff will be able to manage a diary on this site. They can create new notes where they can privately store whatever information that they want to, for example, their thoughts throughout the day, creative writing, to do lists, or upcoming deadlines. They can also delete these notes if necessary. Diary management is a flexible platform for everyone related to the university, with the goal being not to serve only academic purposes but also expressive freedom.

* Report Generation/ Management:

Reports can be generated by admins as well as the teaching staff. They can be used to schedule meetings on different topics such as not attending a personal tutorial meeting, having missed any classes, for missing assignments and so on. Fields such as student’s name, and the reason for report generation is to be filled and the meeting date is also to be fixed. In this way reports can be created, after which a report with the concerning subject is addressed to the student which can also be printed.

#### 2.2.2.1 Access Rights:

The following tables display different features of various areas and the rights that the key user groups, admin, staff and student possess on the matter of using these features.

* Student record management:

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Administration** | **Staff** | **Student** |
| Add student record | Add new record | No access | No access |
| Edit student record | Update existing records | No access | No access |
| Display student record | Show all existing records | Can view basic details when grading assignments and if they are the student’s personal tutor | Own profile |
| Assign students to a course | Allocate student to course of their choice | No access | No access |
| Archive student record | Archive student record | No access | No access |
| Unarchive student record | Unarchive student record | No access | No access |
| Delete student record | Delete existing record | No access | No access |

* Staff record management:

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Administration** | **Staff** | **Student** |
| Add staff record | Add new record | No access | No access |
| Edit staff record | Update existing records | No access | No access |
| Display staff  record | Show existing records | Own profile | View basic information of their lecturers and personal tutors |
| Assign staff | Allocate staff to different job positions | Can view their job position on their profile | No access |
| Archive staff record | Archive staff record | No access | No access |
| Unarchive staff record | Unarchive staff record | No access | No access |
| Delete staff record | Delete existing record | No access | No access |

* Course Management:

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Administration** | **Staff** | **Student** |
| Add course | Add new course | No access | No access |
| Edit course details | Update information of existing course | No access | No access |
| Display course details | Show details of existing course | No access | Can view enrolled course |
| Delete course | Remove course | No access | No access |
| Assign course leader | Add a course leader | No access | No access |
| Archive course record | Archive course record | No access | No access |
| Unarchive course record | Unarchive course record | No access | No access |

* Module Management:

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Administration** | **Staff** | **Student** |
| Add module record | Add new module | No access | No access |
| Add modules to a course | Add modules to a course | No access | No access |
| Edit module details | Update information of existing module | No access | No access |
| Display module details | Show details of existing module | Can view modules they teach on their profile | Can view modules under enrolled course |
| Delete module | Remove module | No access | No access |
| Assign module leader | Add module leader | Can view module on their profile | Can view module leaders of the modules they are taking |
| Archive module record | Archive module record | No access | No access |
| Unarchive module record | Unarchive module record | No access | No access |

* Assignment Management:

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Administration** | **Staff** | **Student** |
| Add assignment | No access | Add new assignment | No access |
| Edit assignment details | No access | Update information of assignment | No access |
| Display assignment details | No access | Show assignment details | View details and download files(if attached) of respective assignment |
| Assign assignment | No access | Assignments assigned to the module | No access |
| Delete assignment | No access | Remove assignment | No access |
| Submit Assignment | No access | No access | Can submit assignments of modules they are taking |
| Grade Assignment | No access | Can view student submissions and provide grades with feedback | No access |
| Display assignment grades | No access | No access | Can view grades for submitted assignment |
| Archive assignment record | No access | Archive assignment record | No access |
| Unarchive assignment record | No access | Unarchive assignment record | No access |

* Attendance Records Management:

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Administration** | **Staff** | **Student** |
| Add attendance | Add attendance | Add attendance | No access |
| Edit attendance | Can edit attendance | Edit attendance | No access |
| Display attendance list | Can view attendance | Can view attendance | Can view their own attendance records |
| Action for poor attendance | Send mail to student | Send mail to student | No access |
| Delete attendance | Remove attendance | Remove attendance | No access |
| Archive attendance record | Archive attendance record | Archive attendance record | No access |
| Unarchive attendance record | Unarchive attendance record | Unarchive attendance record | No access |
| Search records by student name or email | Can search records of a particular student | Can search records of a particular student | No access |

* Personal Tutor Management:

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Administration** | **Staff** | **Student** |
| Add personal tutor | Add personal tutor | No access | No access |
| Edit personal tutor | Edit personal tutor | No access | No access |
| Display personal tutor | Can view list of available personal tutors | Can view group of students who have the teacher assigned as their personal tutor | Can view information of their own personal tutor only |
| Assign personal tutor | Assign personal tutor to students | No access | No access |

* Timetable Management:

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Administration** | **Staff** | **Student** |
| Add timetable | Add timetable record | Add timetable record | No access |
| Edit timetable | Edit timetable record | Edit timetable record | No access |
| Display timetable | View all timetable records | View all timetable records | Can view timetable containing modules they take |
| Delete time table | Remove timetable record | Remove timetable record | No access |
| Archive timetable record | Archive timetable record | Archive timetable record | No access |
| Unarchive timetable | Unarchive timetable record | Unarchive timetable record | No access |

* Diary Management:

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Administration** | **Staff** | **Student** |
| Add note | Can add a new personal note | Can add a new personal note | Can add a new personal note |
| Edit note | Edit personal notes | Edit personal notes | Edit personal notes |
| Display note | Can view all personal notes as well as details of individual notes | Can view all personal notes as well as details of individual notes | Can view all personal notes as well as details of individual notes |
| Delete note | Can delete their personal notes | Can delete their personal notes | Can delete their personal notes |

* Report Management:

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Administration** | **Staff** | **Student** |
| Create report | Can generate personalized report based on reason and student name. | Can generate personalized report based on reason and student name. | No access |
| Display report | Can view generated report | Can view generated report | No access |
| Print report | Can print report | Can print report | No access |

#### 2.2.2.2 External Output Format:

External output formats are shown below which are used to send notifications to the users through e-mails.

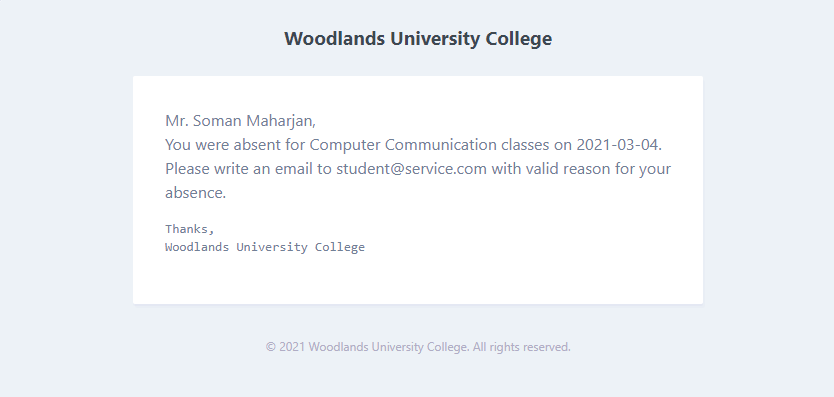
Outline for Student’s attendance report email:

Mr. /Mrs. *Title Name,*

You were absent for *Module Name* classes on *Date.*  
Please write an email to student@service.com with valid reason for your absence.

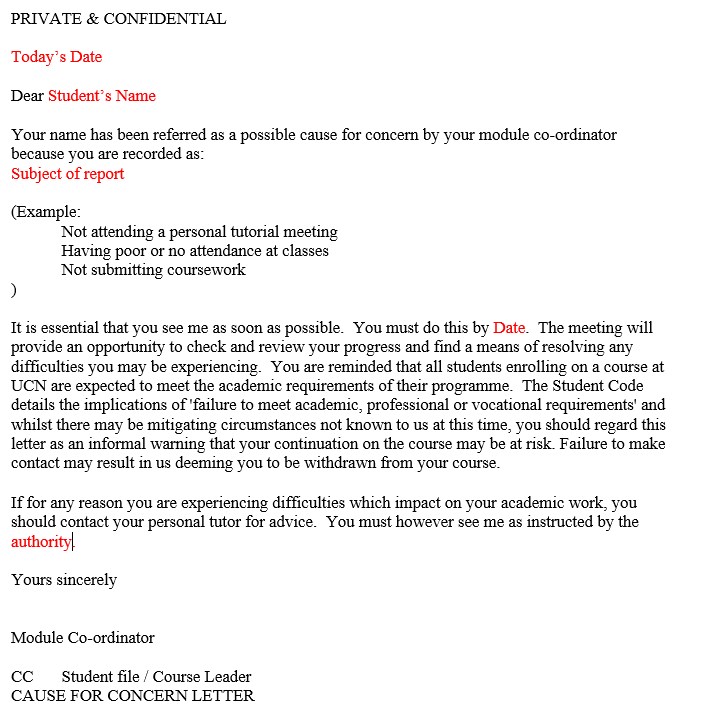
Thanks,

Woodlands University College



Outline for displaying and printing reports addressed to a student regarding issues such as missing classes or assignments, is given below:

Example:



### **2.2.3 Performance** **Requirements:**

Performance Requirements in requirement specification comprises of requirements which specifies how the system should work based on factors such as speed, cost, capacity, appearance, durability, strength, security, privacy and so on to support end user tasks. Some of the performance requirement constraints are discussed below:

#### 2.2.3.1 Speed

* Throughput: Throughput is the rate at which the system is proceed. The new throughput will be faster than current scenario.
* Response Time: It is the amount of time taken for the system to react on a given input. We are estimating for it to enact around at least 7 seconds. In case of buffers, the time shall last a bit longer.

#### 2.2.3.2 Capacity

Capacity amount accords with the numeric figure of something a particular system can hold. This is an important factor since it broadly affects how the system will be used. The capacity needs of the university will be considered for the system. It will be able to support the record of around 3000 individuals including teachers, students and administrative authorities with all their respective records. Moreover, the system will be made as flexible as possible.

#### 2.2.3.3 Reliability

Reliability denotes the probability that a system will operate without failure. The system shall run 24/7 with constant checks on maintenance to avoid potential bugs.

#### 2.2.3.4 Usability

Usability includes qualities of a system which makes user experience better. The system will have a simple and understandable layout. It will run smoothly on all Medias including mobile phones. The functionalities of the system will be displayed the navigation panel. Overall, the system will be more user-friendly, efficient, error tolerant and easy to use.

### **Design Constraints:**

The details below include the design constraints that were decided for this project based on different factors, a major factor being the client elicitation interviews. Constraints keeping in mind different functional and non-functional requirements are described below.

**Operating System:**

As Windows and Linux are most commonly used operating systems, the application is designed to be compatible with both OS but not limited to them. It is also compatible with any operating system that can run package managers like composer and npm.

**Front-end graphics:**

There are no limitations posed by the client in terms of color schemes for the site, so we decided to go with light colors for the background and darker colors for the navigation and column names. For buttons, colors like red, green and blue are to be used where red is for deleting data, blue is for editing and so on. Appropriate icons are also to be used so that the site is easy to use from the user's end. The logo of the university is to be presented on all pages of the site, including the login page. The design is to be made responsive so that it is can be displayed and used with ease in different screen sizes.

There will be three user groups that are going to use the system so the options on the navigation is going to depend upon the access rights of these users. The goal is to use a consistent and minimalistic design for the site which makes it user-friendly and readable. The application will be heavily focused on functionality, so the idea of adding animations and images has been rejected and the application will have default bootstrap icons, navigation bar, forms and buttons.

**Hardware and programming languages:**

Hardware requirements include that the system is to be used for desktops. Furthermore, as there were no set constraints about the programming languages to be used, the team agreed to work with PHP. The site will be developed using Laravel, a PHP framework and follow MVC (Model View Controller) architecture. HTML, CSS, and JavaScript, Bootstrap will also be used for the front-end design.

**Database:**

MySQL will be used to store all the data related to the system.

### **Commercial Constraints:**

This shows the breakdown of the total budget in various parts of the development. The budget was not specified, which is why we chose to reserve a total of $20,000 for this project.

|  |  |  |  |
| --- | --- | --- | --- |
| **Project Information** | | | |
| Total length of the project: | 20 weeks |  |  |
| Hourly wage: | £15 |  |  |
| Number of workers: | 4 |  |  |
| **Project Breakdown:** | | | |
| Sections | Week | Hours per week  Per person | Total Wage |
| Requirement Specification | 4 | 7 | £750.00 |
| Design and Analysis | 5 | 7 | £1000.00 |
| **Project build breakdown** | 10 |  |  |
| Console | 25% | 20 | £1000.00 |
| Website | 30% | 20 | £1500.00 |
| Mobile Application | 10% | 20 | £1500.00 |
| Testing | 30% | 20 | £1500.00 |
| Evaluation | 5% | 20 | £1500.00 |
| **Additional Cost** | | | |
| **Reason** | | **Cost** | |
| Software/Hardware | | £3000.00 | |
| Office Costs | | £1000.00 | |
| **Total Build Cost** | | | |
| Other expenses | | £1000.00 | |
| Profit Margin | | £3000.00 | |
| Program rights | | Negotiable | |
|  | | | |
| **Total Project Cost** | | £16750.00 | |

### **Acceptance Tests:**

Acceptance Testing is conducted prior to making the system available for actual use for the client, to make sure that the system can run smoothly without any possible errors. The system is to be tested to make sure that all it fulfills all the requirements that were put forward by the client. Types of acceptance testing to be carried out is given as follows:

Black Box Testing:

This kind of testing is to be carried out to test all the functionalities of the system. By taking the documents of requirements provided by the client, and the findings from client elicitation interviews, the system is to be tested to check whether all requirements are fulfilled and whether they work as intended. Checking whether the system is easy to use and if it the performance is as expected is also done here. To carry out black box testing, checking the internal structure of the code is not necessary as this method is used for monitoring only the features of the system meant for the end users. Testing will be done by creating different scenarios by providing both valid and invalid inputs and check if the actual outputs match expected outputs.

# System Analysis and Design:

System Analysis and Design is conducted after analyzing the problem domain, to simplify and categorize a system’s functionality into different components. This is done to take a closer look into the functionalities to be included into the system.

## Preliminary Design Stages:

### **Textual Analysis:**

|  |  |
| --- | --- |
| **Candidate Classes** | **Responsibilities** |
| **Admin** | * Add/ edit/ delete staff record * Archive/ unarchive staff record * Display all existing staff records * Add/ edit/ delete student record * Assign students to a course * Archive/ unarchive student record * Display all existing student records |
| **Staff** | * Post assignments (to students taking the module) * Grade assignments * Display graded assignments * View personal profile |
| **Student** | * View posted assignments * Submit Assignments * Display grades of submitted assignments * View personal profile * View timetable * Create Note * Delete Note * View timetable |
| **Course** | * Add/ edit/ delete course record * Display all courses * Archive/ unarchive course record * Assign course leader |
| **Module** | * Add/edit/delete module record * Add module to course * Assign module leader * Display all modules * Archive/ unarchive module record |
| **Attendance** | * Add/ edit/ delete attendance record * Archive/unarchive attendance record * Display all attendance records * Report poor attendance record of a student * Search for attendance records of a student |
| **Personal Tutor** | * Add personal tutor * Edit personal tutor * Display all available personal tutors * Assign personal tutor |
| **Diary** | * Add/ edit/delete note * Display all notes |
| **Report** | * Create report record * Display report * Print report |
| **Assignment** | * Add/ edit/ delete assignment record * Display all assignments * Archive/ unarchive assignment record |
| **Timetable** | * Add/ edit/ delete timetable record * Display timetable records * Archive/ unarchive timetable records |

### **Significant Event Analysis:**

|  |  |  |
| --- | --- | --- |
| **Event** | **Performers** | **Candidate Attributes** |
| **Login** | * Admin * Student * Staff | * user\_id * role\_id * email * password |
| **Display all courses’ record** | * Admin | * user\_id * role\_id * email * password * course\_id * course\_name * description * credit\_score * cost * start\_date * end\_date * semester * is\_archived |
| **Display enrolled course record** | * Student | * user\_id * role\_id * email * password * course\_id * course\_name * description * credit\_score * cost * start\_date * end\_date * is\_archived * semester * course\_name |
| **Add/Edit/delete course record** | * Admin | * user\_id * role\_id * email * password * course\_id |
| **Archive/Unarchive course record** | * Admin | * user\_id * role\_id * email * password * course\_id * is\_archived |
| **Assign course leader** | * Admin | * user\_id * role\_id * email * password * course\_id * course\_leader |
| **View own profile** | * Staff * Student | * user\_id * role\_id * email * password * first\_name * last\_name * date\_of\_birth * mobile\_number * email * contact\_number * date\_joined |
| **Display all assignment record** | * Staff | * user\_id * role\_id * email * password * assignment\_id * title * description * module\_id * assignment\_file * submission\_date * is\_archived |
| **Add/edit/delete Assignment**  **record** | * Staff | * user\_id * role\_id * email * password * assignments\_id * assignments\_title |
| **Display all student record** | * Admin | * user\_id * role\_id * email * password * student\_first\_name * student\_middle\_name * student\_last\_name * student\_id * address * student\_email * contact\_number * course\_id * is\_archived |
| **Add/edit/delete student record** | * Admin | * user\_id * role\_id * email * password * student\_id * student\_first\_name * student\_middle\_name * student\_last\_name |
| **Archive/ unarchive student**  **record** | * Admin | * user\_id * role\_id * email * password * student\_id * is\_archived |
| **Assign student to a course** | * Admin | * user\_id * role\_id * email * password * course\_id * course\_name * student\_id |
| **Add/edit/delete staff record** | * Admin | * user\_id * role\_id * email * password * staff\_id * staff\_first\_name * staff\_middle\_name * staff\_last\_name |
| **Archive/ unarchive staff** | * Admin | * user\_id * role\_id * email * password * staff\_id * is\_archived |
| **Display all staff record** | * Admin | * user\_id * role\_id * email * password * staff\_id * staff\_first\_name * staff\_middle\_name * staff\_last\_name * email * contact\_number * is\_archived |
| **Assign staff to job position** | * Admin | * user\_id * role\_id * email * password * job\_title * staff\_id |
| **View job position** | * Staff * Admin | * user\_id * role\_id * email * password * job\_title * staff\_id |
| **View all student records** | * Admin | * user\_id * role\_id * email * password * student\_first\_name * student\_middle\_name * student\_last\_name * student\_id * contact\_number * student\_email * course\_id * is\_archived |
| **Add/ edit/ delete module record** | * Admin | * user\_id * role\_id * email * password * module\_id * module\_name |
| **Add module to course** | * Admin | * user\_id * role\_id * email * password * module\_id * module\_name * course\_id * course\_name |
| **Assign module leader** | * Admin | * user\_id * role\_id * email * password * module\_id * staff\_id * staff\_first\_name * staff\_middle\_name * staff\_last\_name |
| **Display all module records** | * Admin | * user\_id * role\_id * email * password * module\_id * module\_name * module\_code * start\_date * description * end\_date * credit\_score * module\_leader * is\_archived |
| **Display enrolled/teaching module record** | * Staff * Student | * user\_id * role\_id * email * password * module\_id * module\_name * module\_code * start\_date * description * end\_date * credit\_score * module\_leader * is\_archived |
| **Archive/Unarchive module record** | * Admin | * user\_id * role\_id * email * password * module\_id * is\_archived |
| **View respective assignment** | * Student | * user\_id * role\_id * email * password * assignment\_id * assignment\_title * assignment\_file * submission\_date * module\_id |
| **Assign assignments to module** | * Staff | * user\_id * role\_id * email * password * assignment\_id * assignment\_title * module\_id * assignment\_file * submission\_date |
| **Grade assignment** | * Staff | * user\_id * role\_id * email * password * student\_id * student\_assignment * created\_at * grade * feedback * module\_id * title |
| **Display assignment grades** | * Student | * user\_id * role\_id * email * password * assignment\_grade\_id * module\_id * report\_id * grade * feedback |
| **Archive/ unarchive assignment records** | * Staff | * user\_id * role\_id * email * password * assignment\_id * is\_archived |
| **Add/ edit/ delete attendance record** | * Admin * Staff | * user\_id * role\_id * email * password * module\_id * student\_id * first\_name * middle\_name * last\_name * attendance\_date * status * module\_name |
| **Display attendance record** | * Admin * Staff | * user\_id * role\_id * email * password * module\_id * student\_id * first\_name * middle\_name * last\_name * attendance\_date * status * module\_name * is\_archived |
| **Archive/ unarchive attendance record** | * Admin * Staff | * user\_id * role\_id * email * password * attendance\_id * module\_id * student\_id * is\_archived |
| **Search attendance records by student name or email** | * Admin * Staff | * user\_id * role\_id * email * password * student\_id * student\_first\_name * student\_middle\_name * student\_last\_name * student\_email |
| **Add/ edit/ delete personal tutor** | * Admin | * user\_id * role\_id * email * password * staff\_id * student\_id |
| **Display all available personal tutor records** | * Admin | * user\_id * role\_id * email * password * staff\_id * staff\_first\_name * staff\_middle\_name * staff\_last\_name |
| **View PAT information** | * Student | * user\_id * role\_id * email * password * staff\_id * staff\_first\_name * staff\_middle\_name * staff\_last\_name * email * contact\_number |
| **View student under PAT** | * Staff | * user\_id * role\_id * email * password * student\_id * student\_first\_name * student\_middle\_name * student\_last\_name * email * contact\_number * pat\_id |
| **Assign PAT** | * Admin | * user\_id * role\_id * email * password * staff\_id * student\_id * pat\_id |
| **Add/ edit/ delete timetable** | * Admin * Staff | * user\_id * role\_id * email * password * timetable\_id |
| **Archive/ unarchive timetable** | * Admin * Staff | * user\_id * role\_id * email * password * timetable\_id * is\_archived |
| **Display all timetable** | * Admin * Staff | * user\_id * role\_id * email * password * timetable\_id * start\_time * end\_time * day * module\_id * timetable\_is\_archived * is\_archived |
| **Display personal timetable** | * Student | * user\_id * role\_id * email * password * timetable\_id * start\_time * end\_time * day * module\_id |
| **Add/ edit/ delete note** | * Admin * Staff * Student | * user\_id * role\_id * email * password * note\_id |
| **Display note** | * Admin * Staff * Student | * user\_id * role\_id * email * password * note\_id * user\_id * title * note |
| **Create report** | * Admin * Staff | * user\_id * role\_id * email * password |
| **Display report** | * Admin * Staff | * user\_id * role\_id * email * password |
| **Print report** | * Admin * Staff | * user\_id * role\_id * email * password |

## Detailed Statistic System Designs:

### **BON Class Charts:**

|  |  |  |  |
| --- | --- | --- | --- |
| **CLASS** | *COURSE* | | **Part:** 1/1 |
| **TYPE OF OBJECT**  Represents all courses offered by the university. | |  | |
| **Queries** | user\_id, role\_id, email, password, course\_name, semester, course\_escription, credit\_score, course\_start\_date, course\_end\_date, course\_is\_archived, course\_cost, course\_leader, course\_id | | |
| **Commands** | * Add/ edit/ delete course record * Display all courses * Archive/ unarchive course record * Assign course leader | | |
| **Constraints** | Cannot edit/delete/display/archive a course that does not exist  Cannot unarchive a course that is not archived in the first place  Cannot leave field null | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CLASS** | *STAFF* | | **Part:** 1/1 |
| **TYPE OF OBJECT**  Represents all staff employed  by the university | |  | |
| **Queries** | user\_id, role\_id, email, password, staff\_id, staff\_first\_name, staff\_middle\_name, staff\_last\_name, staff\_address, staff\_date\_of\_birth, staff\_date\_joined, staff\_email,gender, staff\_contact\_number, salary, job\_title, staff\_date\_joined, staff\_is\_archived | | |
| **Commands** | * Post assignments (to students taking the module) * Grade assignments * Display graded assignments * View personal profile | | |
| **Constraints** | Cannot grade assignments that were not submitted  Cannot leave fields null | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CLASS** | *STUDENT* | | **Part:** 1/1 |
| **TYPE OF OBJECT**  Represents all students enrolled in  the university | |  | |
| **Queries** | user\_id, role\_id, email, password, student\_id, student\_first\_name, student\_middle\_name, student\_last\_name, student\_address, student\_date\_of\_birth, student\_date\_joined, student\_email, student\_gender, student\_contact\_number, pat\_id, course\_id, sudent\_is\_archived | | |
| **Commands** | * View posted assignments * Submit Assignments * Display grades of submitted assignments * View personal profile * View timetable * Create Note * Delete Note * View timetable | | |
| **Constraints** | Cannot edit/delete/display a note that does not exist  Cannot submit/view assignment that does not exist  Cannot display timetable that does not exist  Cannot leave field null | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CLASS** | *ADMIN* | | **Part:** 1/1 |
| **TYPE OF OBJECT**  Represents all staff employed  by the university | |  | |
| **Queries** | user\_id, role\_id, email, password, staff\_id, staff\_first\_name, staff\_middle\_name, staff\_last\_name, staff\_address, staff\_date\_of\_birth, staff\_date\_joined, staff\_email,gender, staff\_contact\_number, salary, job\_title, staff\_date\_joined, staff\_is\_archived | | |
| **Commands** | * Add/ edit/ delete staff record * Archive/ unarchive staff record * Display all existing staff records * Add/ edit/ delete student record * Assign students to a course * Archive/ unarchive student record * Display all existing student records | | |
| **Constraints** | Cannot edit/delete/display/archive a staff record that does not exist  Cannot edit/delete/display/archive a student record that does not exist  Cannot add a student record that already exists  Cannot unarchive a staff record that is not archived in the first place  Cannot unarchive a student record that is not archived in the first place  Cannot leave field null | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CLASS** | *MODULE* | | **Part:** 1/1 |
| **TYPE OF OBJECT**  Represents all modules taught in the university | |  | |
| **Queries** | user\_id, role\_id, email, password, module\_id, module\_name, module\_start\_date, module\_description, module\_end\_date, creadit\_score, module\_leader, module\_is\_archived | | |
| **Commands** | * Add/edit/delete module record * Add module to course * Assign module leader * Display all modules * Archive/ unarchive module record | | |
| **Constraints** | Cannot edit/delete/display/archive a module that does not exist  Cannot unarchive a module that is not archived in the first place  Cannot assign a module leader to a module that does not exist  Cannot assign a module leader who is not a staff  Cannot leave field null | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CLASS** | *ASSIGNMENT* | | **Part:** 1/1 |
| **TYPE OF OBJECT**  Represents all assignments created by tutors | |  | |
| **Queries** | user\_id, role\_id, email, password, assignment\_id,assignment\_title, assignment\_description, assignment\_file, submission\_date, module\_id, assignment\_is\_archived | | |
| **Commands** | * Add/ edit/ delete assignment record * Display all assignments * Archive/ unarchive assignment record | | |
| **Constraints** | Cannot edit/delete/display/archive an assignment record that does not exist  Cannot unarchive an assignment record that is not archived in the first place  Cannot leave field null | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CLASS** | *ATTENDANCE* | | **Part:** 1/1 |
| **TYPE OF OBJECT**  Represents all staff employed  by the university | |  | |
| **Queries** | user\_id, role\_id, email, password, student\_id, module\_id, status, attendance\_date, attendance\_is\_archived | | |
| **Commands** | * Add/ edit/ delete attendance record * Archive/unarchive attendance record * Display all attendance records * Report poor attendance record of a student * Search for attendance records of a student | | |
| **Constraints** | Cannot edit/delete/display/archive an attendance record that does not exist  Cannot unarchive an attendance record that is not archived in the first place  Cannot search for student that is not enrolled  Cannot leave field(status i.e. present or absent) null | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CLASS** | *DIARY* | | **Part:** 1/1 |
| **TYPE OF OBJECT**  Represents personal notes created by admin, staff or students | |  | |
| **Queries** | user\_id, role\_id, email, password, note\_id, title, note | | |
| **Commands** | * Add//delete note * Display all notes | | |
| **Constraints** | Cannot edit/delete/display a diary record that does not exist  Cannot leave fields (title or note) null | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CLASS** | *REPORT* | | **Part:** 1/1 |
| **TYPE OF OBJECT**  Represents all reports submitted by students | |  | |
| **Queries** | user\_id, role\_id, email, password, report\_id, student\_id, module\_id, assignment\_id, student\_assignment, assignment\_title, title\_description | | |
| **Commands** | * Create report record * Display report * Print report | | |
| **Constraints** | Cannot edit/delete/display/print a report that does not exist  Cannot leave field null | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CLASS** | *TIMETABLE* | | **Part:** 1/1 |
| **TYPE OF OBJECT**  Represents timetables, which includes classes of different modules to be taught and attended | |  | |
| **Queries** | user\_id, role\_id, email, password, timetable\_id, start\_time, end\_time, day, module\_id, time\_table\_is\_archived | | |
| **Commands** | * Add/ edit/ delete timetable record * Display timetable records * Archive/ unarchive timetable records | | |
| **Constraints** | Cannot edit/delete/display/archive a timetable that does not exist  Cannot unarchive a timetable that is not archived in the first place | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CLASS** | *PERSONAL TUTOR* | | **Part:** 1/1 |
| **TYPE OF OBJECT**  Represents all staff employed  by the university | |  | |
| **Queries** | user\_id, role\_id, email, password,  pat\_id, student\_id, staff\_id | | |
| **Commands** | * Add personal tutor * Edit personal tutor * Display all available personal tutors * Assign personal tutor | | |
| **Constraints** | Cannot edit/delete/display a personal tutor record that does not exist  Cannot assign personal tutor to students that are not enrolled  Cannot assign personal tutor who is not a staff | | |

## Other Diagrams:

### **Data Flow Diagrams:**

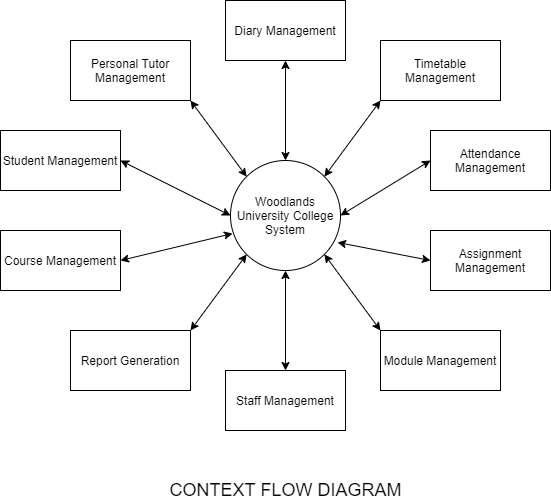
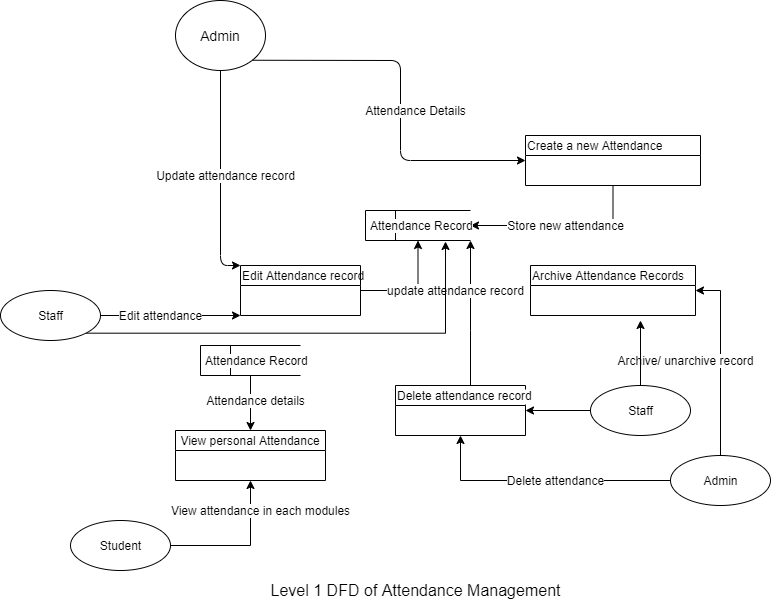
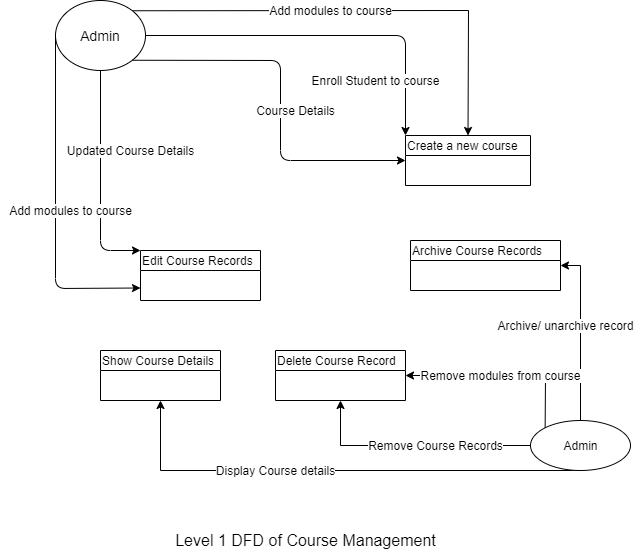
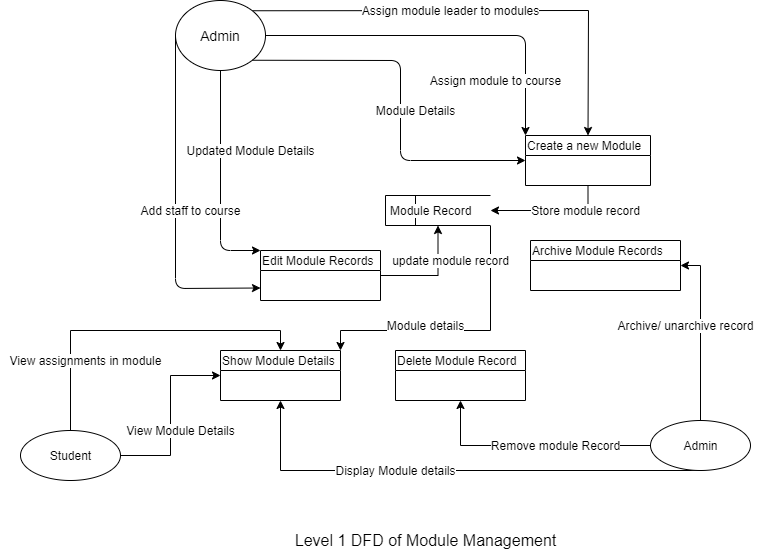
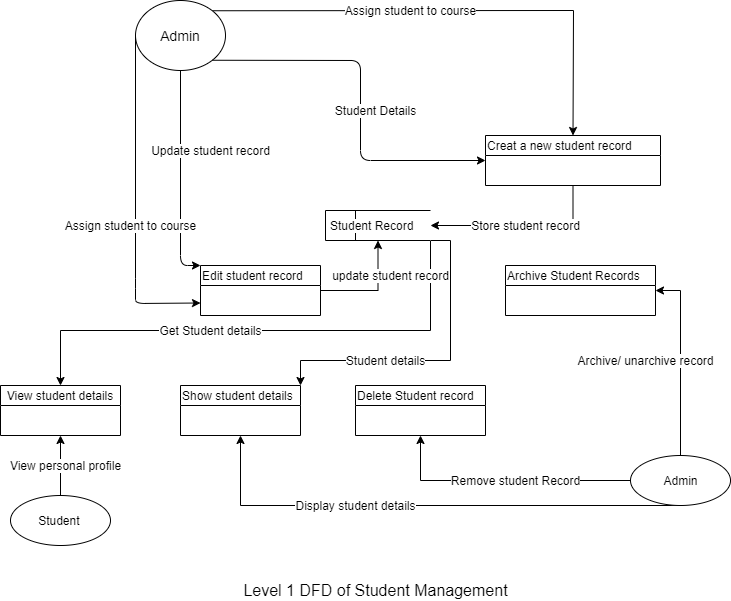
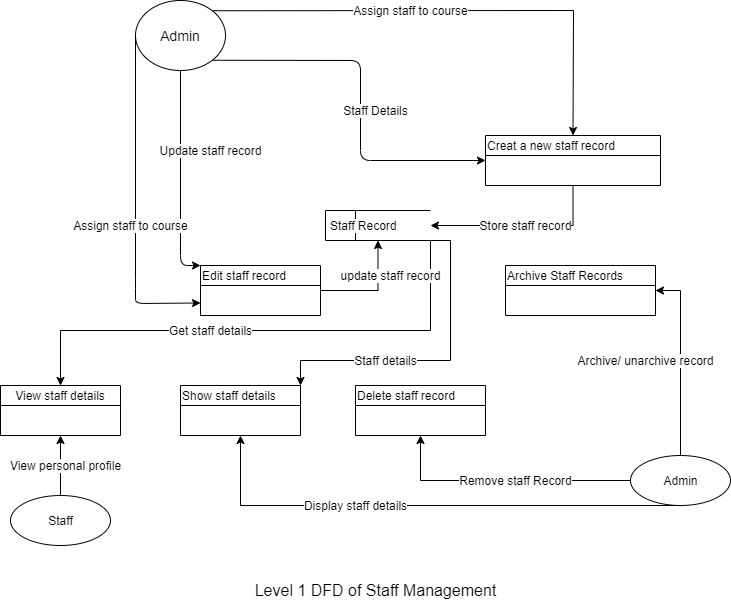
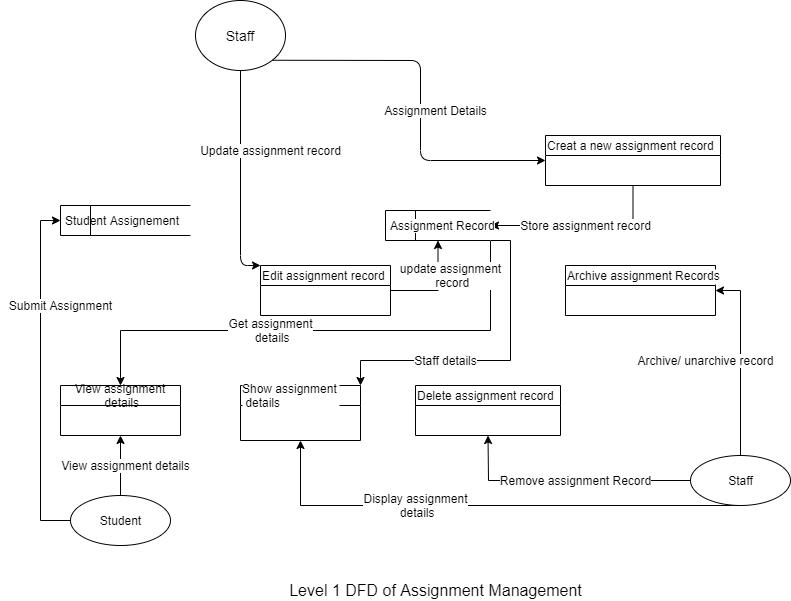
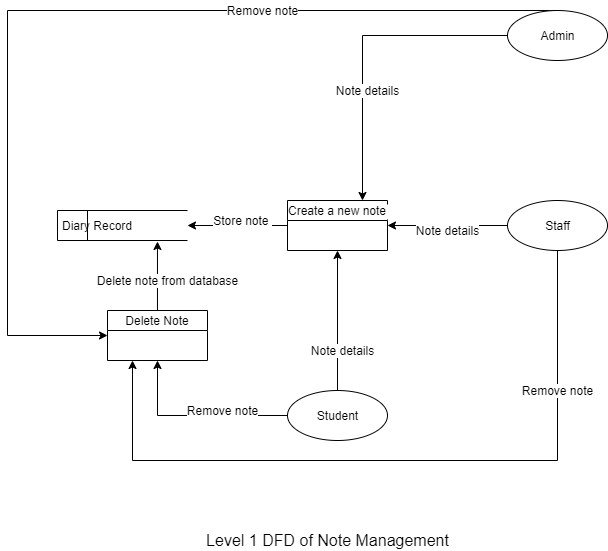
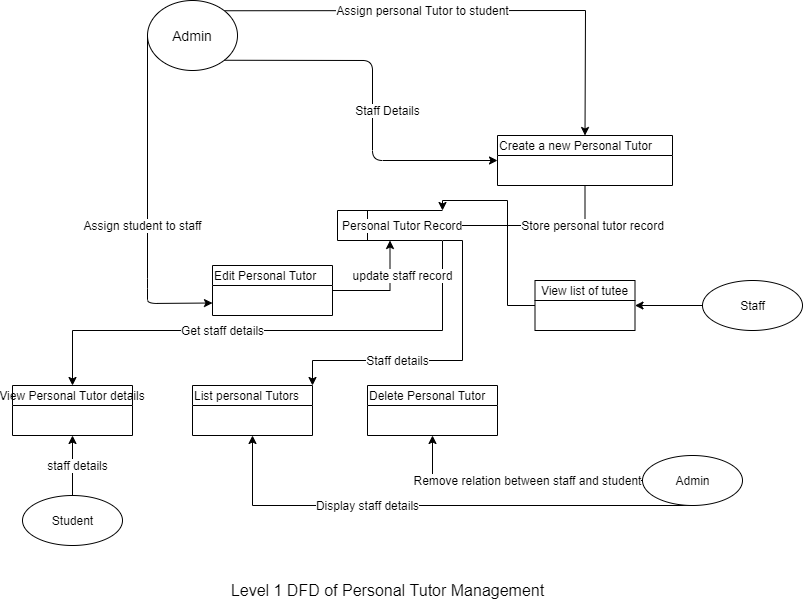
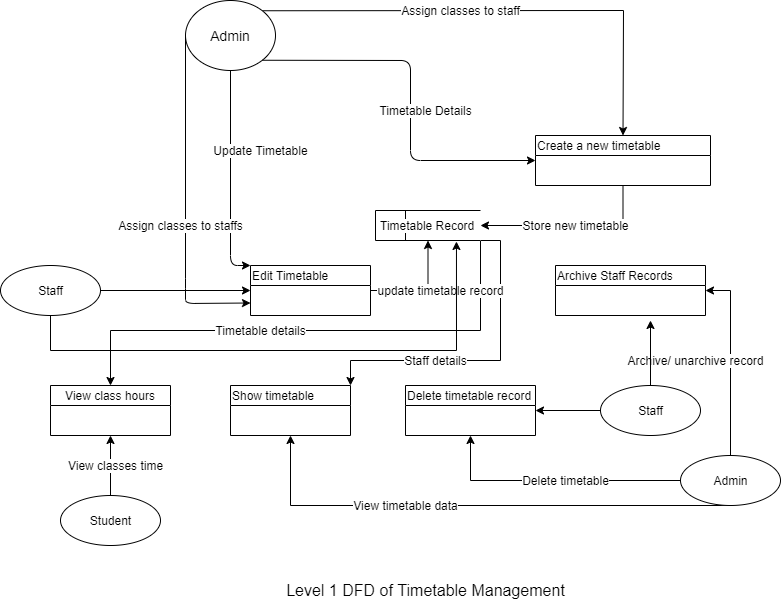
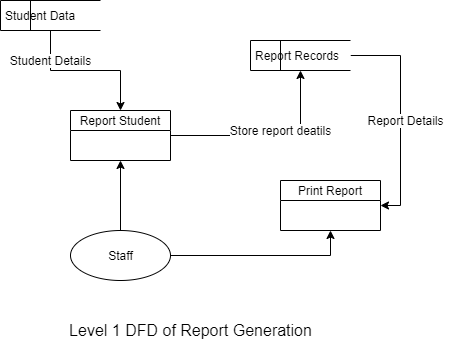
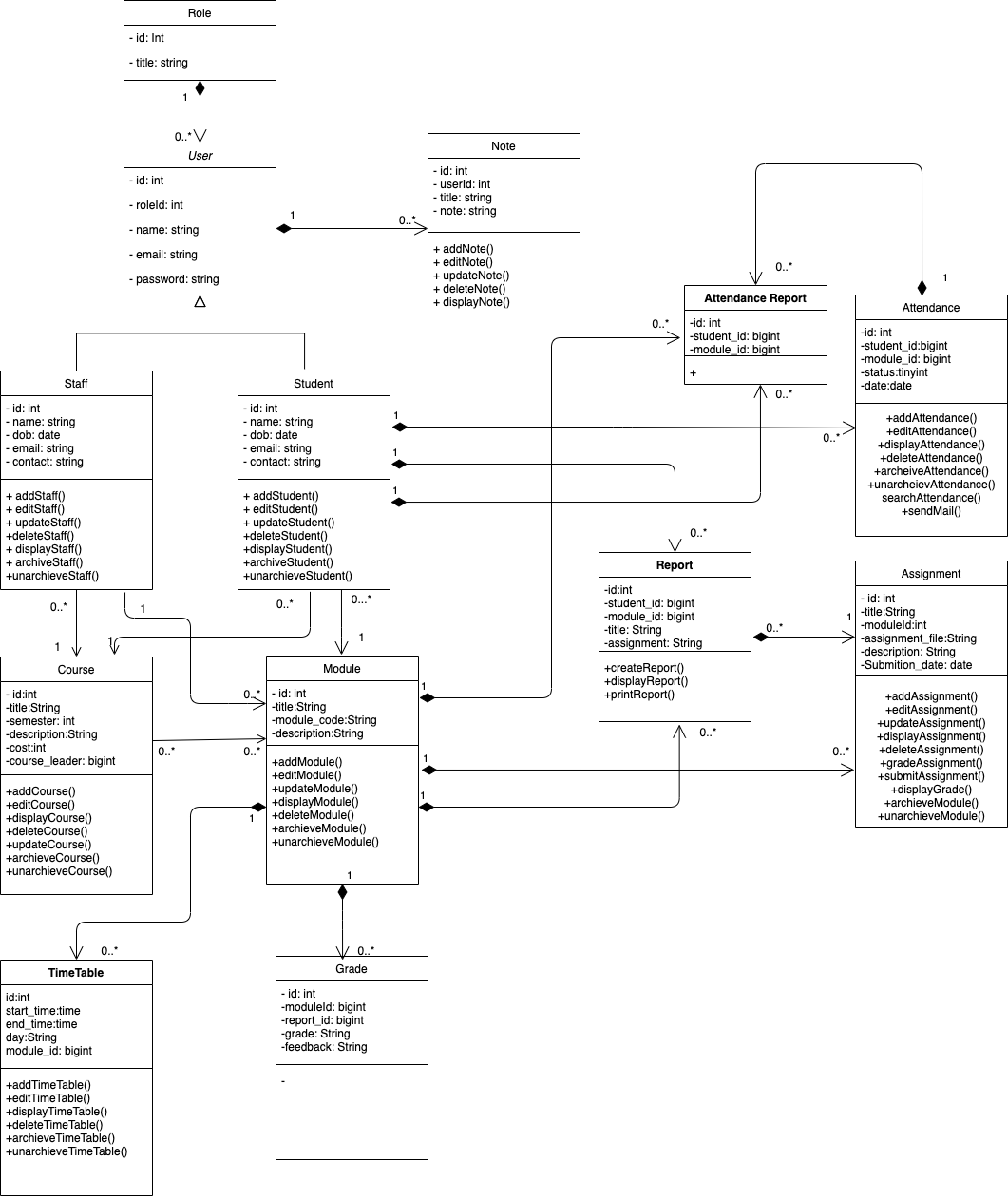


Fig. Level 0 Context Flow Diagram



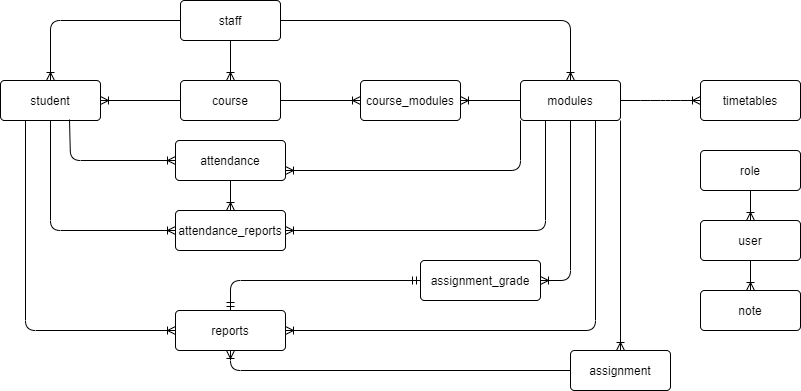


### **Class Diagram:**

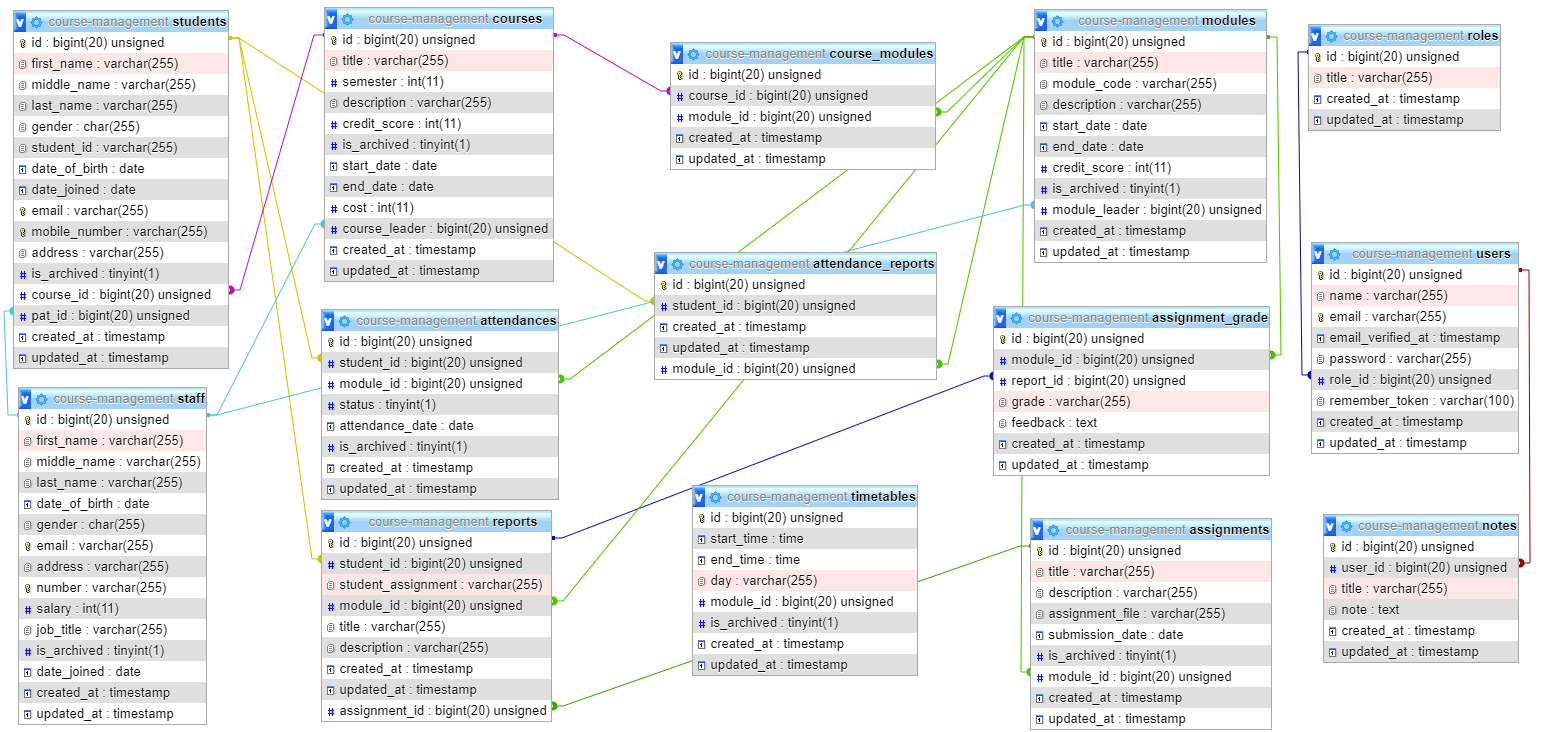


## 3.4 System Database Design:

### **3.4.1 E-R Model:**

****

**Diagram with attributes:**

****

### **3.4.2 Attribute Listings:**

List of different tables with all the attributes used in the table with its data types, constraints and notes are given below:

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Datatype** | **Constraint** | **Notes** |
| **students** |  |  |  |
| id | BIGINT | PK | Auto increment PK for the table |
| student\_id | VARCHAR | PK | Indicates student id numbers of each students studying in the university. |
| first\_name | VARCHAR |  | Refers to first name of the student. |
| middle\_name | VARCHAR |  | Refers to middle name of each students. |
| last\_name | VARCHAR |  | Refers to last name of each students. |
| address | VARCHAR |  | Shows student’s address. |
| date\_of\_birth | DATE |  | Shows students’ date of birth. |
| date\_joined | DATE |  | Shows the date respective student got enrolled. |
| email | VARCHAR |  | Shows respective email address. |
| gender | CHAR |  | Indicates their gender. |
| contact\_number | INT |  | Provides their contact number. |
| pat\_id | BIGINT | FK | Shows id of their respective personal tutor. |
| course\_id | BIGINT | FK | Shows id of respective course they are taking. |
| is\_archived | TINYINT |  | Status of whether the record is archived or not |
| created\_at | TIMESTAMP |  | Timestamp of record creation |
| updated\_at | TIMESTAMP |  | Timestamp of record update |
| **staff** |  |  |  |
| staff\_id | INT | PK | Indicates staff id numbers of each students in the university. |
| first\_name | VARCHAR |  | Refers to first name of the respective staff id numbers. |
| middle\_name | VARCHAR |  | Refers to middle name of each staff. |
| last\_name | VARCHAR |  | Refers to last name of each staff. |
| address | VARCHAR |  | Gives home location of respective staff. |
| date\_of\_birth | DATE |  | Shows staff’ date of birth. |
| date\_joined | DATE |  | Shows the date respective staff got enrolled in the university. |
| email | VARCHAR |  | Shows respective email address of staff. |
| gender | VARCHAR |  | Indicates their gender. |
| contact\_number | INT |  | Provides contact number of each respective staff. |
| salary | INT |  | Gives us the salary information of the respective staff. |
| job\_title | VARCHAR |  | Provides us with the information on their job specifics in the university. |
| is\_archived | TINYINT |  | Status of whether the record is archived or not |
| created\_at | TIMESTAMP |  | Timestamp of record creation |
| updated\_at | TIMESTAMP |  | Timestamp of record update |
| **courses** |  |  |  |
| course\_id | INT | PK | Indicates the course if numbers of each courses studied in the university. |
| name | VARCHAR |  | Gives us the course name of the respective course ids. |
| semester | INT |  | The number of semesters of the course. |
| description | VARCHAR |  | Shows the description of each respective courses. |
| credit\_score | INT |  | Credit score of the course |
| start\_date | DATE |  | Indicates the date where study of course commences. |
| end\_date | DATE |  | Indicates the date where the respective curriculum finishes. |
| is\_archived | TINYINT |  | Status of whether the record is archived or not |
| cost | INT |  | Specifies the cost for entire course curriculum. |
| course\_leader | BIGINT | FK | Shows us the leader for each course assigned. |
| created\_at | TIMESTAMP |  | Timestamp of record creation |
| updated\_at | TIMESTAMP |  | Timestamp of record update |
| **modules** |  |  |  |
| id | BIGINT | PK | Auto increment PK for the table |
| title | VARCHAR |  | Gives us the module name for respective module ids. |
| module\_code | VARCHAR |  | Indicates module code of modules taught in the university. |
| start\_date | DATE |  | Gives us the date of commencement for each module. |
| description | VARCHAR |  | Description of the modules |
| end\_date | DATE |  | Indicates the date where study of the respective modules should complete. |
| credit\_score | INT |  | Specifies the respective credit score. |
| module\_leader | BIGINT |  | Shows us the leader assigned for the respective modules. |
| is\_archived | TINYINT |  | Status of whether the record is archived or not |
| created\_at | TIMESTAMP |  | Timestamp of record creation |
| updated\_at | TIMESTAMP |  | Timestamp of record update |
| **assignments** |  |  |  |
| id | BIGINT | PK | Indicates the assignment id (Auto-increment). |
| title | VARCHAR |  | Indicates the assignment title for respective assignment ids. |
| description | VARCHAR |  | Gives us a brief description for each assignment. |
| file | VARCHAR |  | Name of the file containing the assignment. |
| submission\_date | DATE |  | Date of assignment submission. |
| module\_id | INT | FK | Respective modules of the assignment. |
| is\_archived | TINYINT |  | Status of whether the record is archived or not |
| created\_at | TIMESTAMP |  | Timestamp of record creation |
| updated\_at | TIMESTAMP |  | Timestamp of record update |
| **attendances** |  |  |  |
| id | BIGINT | PK | Indicates the attendance id Auto-increment). |
| student\_id | INT | FK | Indicates the student id numbers for each student. |
| module\_id | INT | FK | Indicates the module id numbers for the respective student enrolled. |
| status | TINYINT |  | Refers to students’ attendance status for that particular module. |
| attendance\_date | DATE |  | Refers to the date of attendance taken. |
| is\_archived | TINYINT |  | Status of whether the record is archived or not |
| created\_at | TIMESTAMP |  | Timestamp of record creation |
| updated\_at | TIMESTAMP |  | Timestamp of record update |
| **notes** |  |  |  |
| id | BIGINT | PK | Indicates the note id (Auto-increment). |
| user\_id | BIGINT | FK | User id of note creator |
| title | VARCHAR |  | Title of the note. |
| note | TEXT |  | Content of the note. |
| created\_at | TIMESTAMP |  | Timestamp of record creation |
| updated\_at | TIMESTAMP |  | Timestamp of record update |
| **reports** |  |  |  |
| report\_id | BIGINT | PK | Refers to the report id numbers(Auto-increment) |
| student\_id | BIGINT | FK | Student id of the student report |
| module\_id | INT | FK | Refers to the module id numbers. |
| assignment\_id | BIGINT | FK | Refers to the assignment id numbers. |
| student\_assignment | VARCHAR |  | Name of the student assignment submission document |
| title | VARCHAR |  | Gives us the report title. |
| description | VARCHAR |  | Gives us a brief description on each reports. |
| created\_at | TIMESTAMP |  | Timestamp of record creation |
| updated\_at | TIMESTAMP |  | Timestamp of record update |
| **timetables** |  |  |  |
| timetable\_id | BIGINT | PK | Refers to the timetable id numbers(Auto-increment) |
| start\_time | TIME |  | Refers to the start time of classes |
| end\_time | TIME |  | Refers to the end time of classes |
| day | VARCHAR |  | Refers to the timetable of a day |
| module\_id | BIGINT | FK | Refers to the module id numbers. |
| is\_archived | TINYINT |  | Status of whether the record is archived or not |
| created\_at | TIMESTAMP |  | Timestamp of record creation |
| updated\_at | TIMESTAMP |  | Timestamp of record update |
| **assignment\_grades** |  |  |  |
| id | BIGINT | PK | Auto increment PK for the table |
| module\_id | INT | FK | Refers to the modules id. |
| report\_id | INT | FK | Refers to the reports for respective modules id. |
| grade | VARCHAR |  | Grade received on the submitted assignment |
| feedback | TEXT |  | Feedback from the teacher on the submitted assignment |
| created\_at | TIMESTAMP |  | Timestamp of record creation |
| updated\_at | TIMESTAMP |  | Timestamp of record update |
| **course\_modules** |  |  |  |
| id | BIGINT | PK | Auto increment PK for the table |
| course\_id | INT | FK | Refers to the course id. |
| module\_id | INT | FK | Refers to the module id present in respective course id. |
| created\_at | TIMESTAMP |  | Timestamp of record creation |
| updated\_at | TIMESTAMP |  | Timestamp of record update |
| **users** |  |  |  |
| id | BIGINT | PK | User id of user |
| name | VARCHAR |  | Name of the user |
| email | VARCHAR |  | Email address of the user |
| password | VARCHAR |  | Password of the user |
| role\_id | BIGINT |  | Role id according to admin, staff or student |
| created\_at | TIMESTAMP |  | Timestamp of record creation |
| updated\_at | TIMESTAMP |  | Timestamp of record update |
| **roles** |  |  |  |
| id | BIGINT | PK | Role id of user role |
| title | VARCHAR |  | Role title of the user |
| created\_at | TIMESTAMP |  | Timestamp of record creation |
| updated\_at | TIMESTAMP |  | Timestamp of record update |
| **attendance\_reports** |  |  |  |
| id | BIGINT | PK | Auto increment PK for the table |
| student\_id | BIGINT | FK | Student id for attendance |
| module\_id | BIGINT | FK | Respective modules for which attendance is taken |
| created\_at | TIMESTAMP |  | Timestamp of record creation |
| updated\_at | TIMESTAMP |  | Timestamp of record update |

# 4 System Interface Designs:

For designing the system interface, wireframes, system navigation diagrams, screen mock-ups, and activity sequence diagrams are used and are included under this section.

## 4.1 Draft Interface Designs:

### **4.1.1 Wireframes:**

The wireframes below show how the basic structure of the site, showing how the login page, home page, displaying forms, and displaying information will look like.

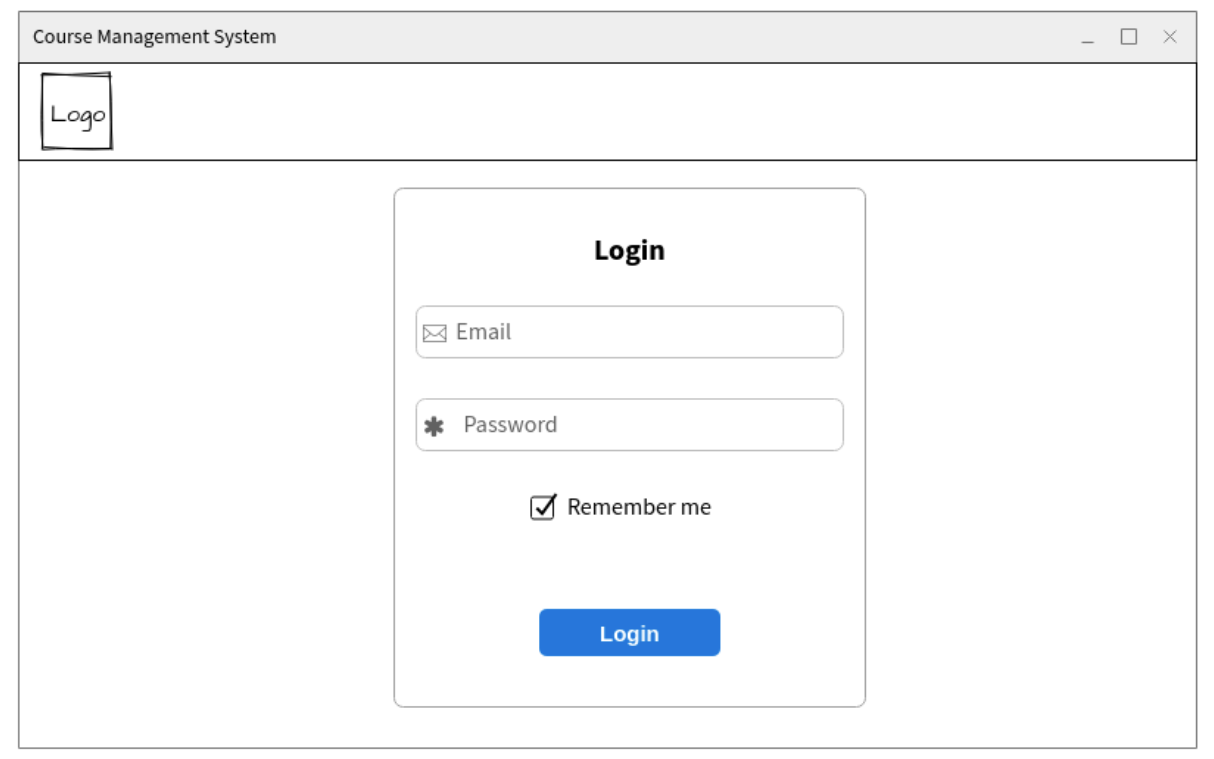
****

Figure 4.1.: Login

****

Figure 4.1.2: Home

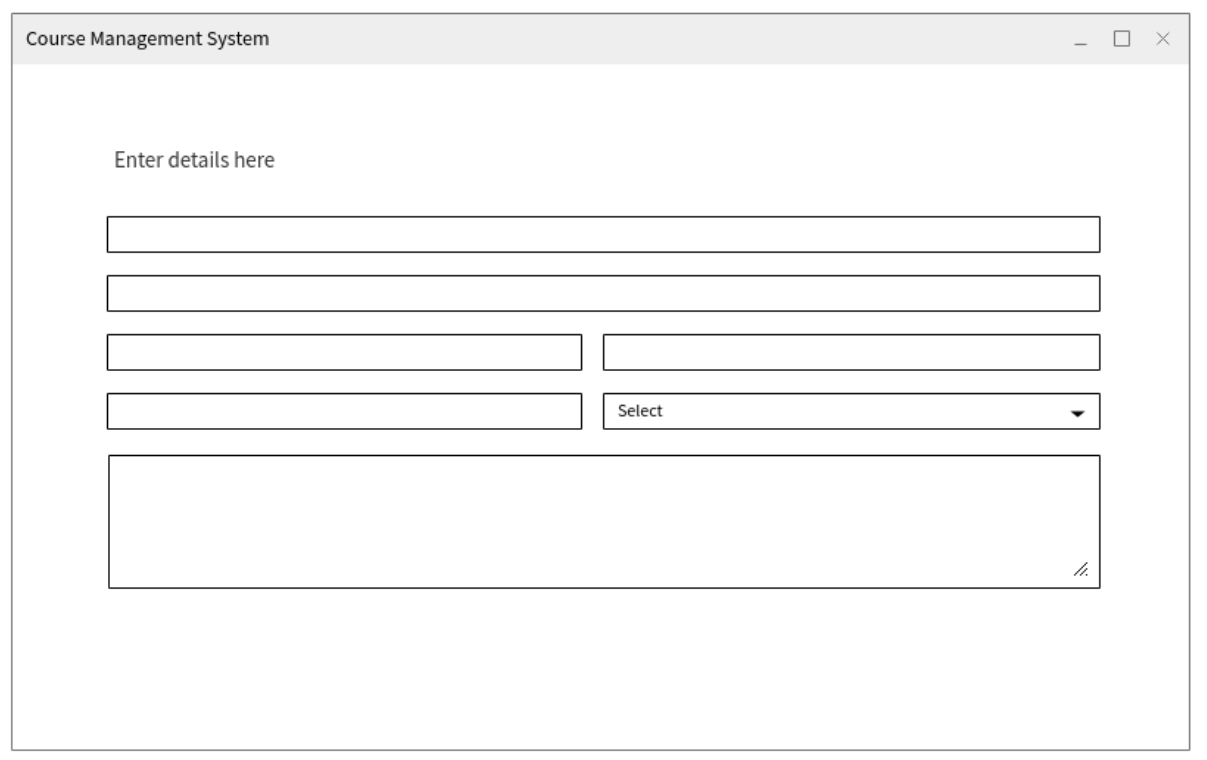
****

Figure 4.1.3: Forms

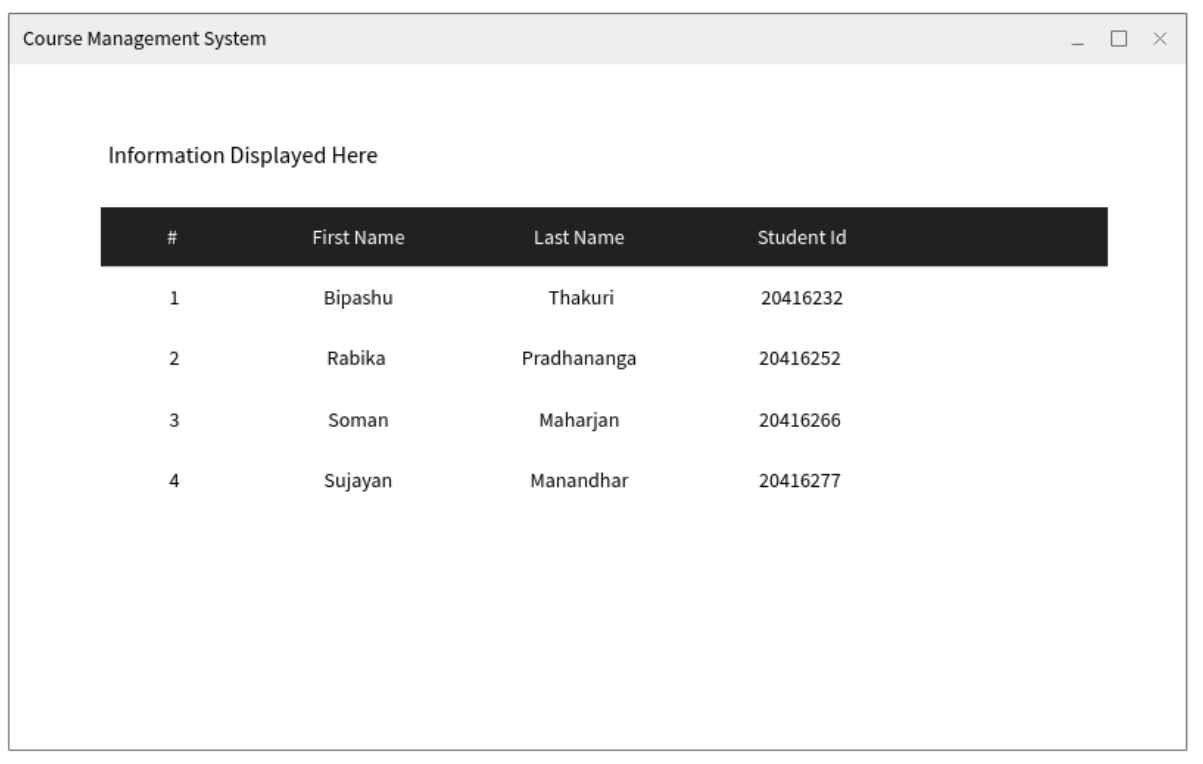
****

Figure 4.1.4: Display of information

### **4.1.2 System Navigation Diagram:**

The following system navigation diagrams show the navigation for different users after they successfully log in to the site. Three different diagrams for admin, staff and student are given below:

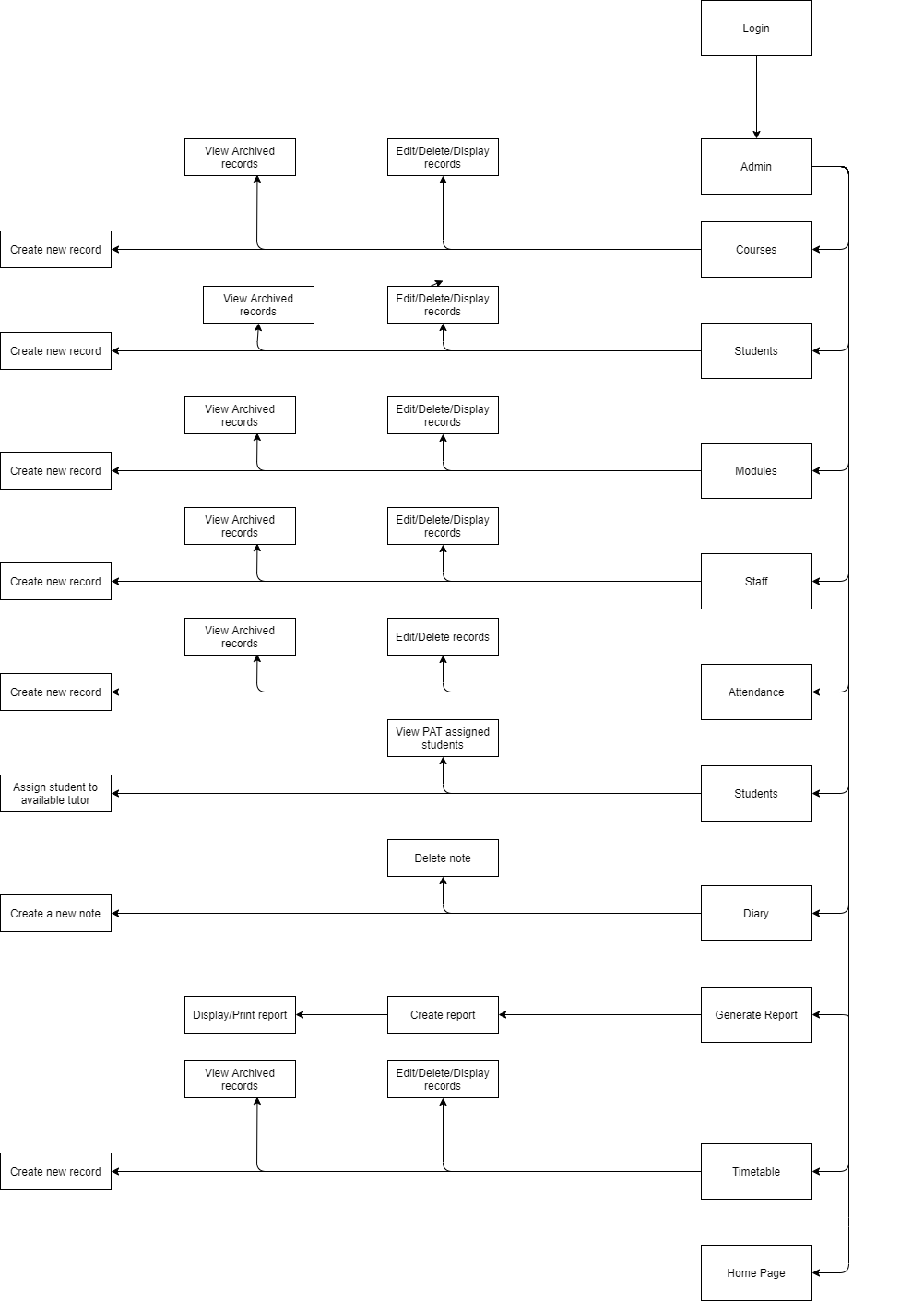


Figure 4.1..1 For Admin

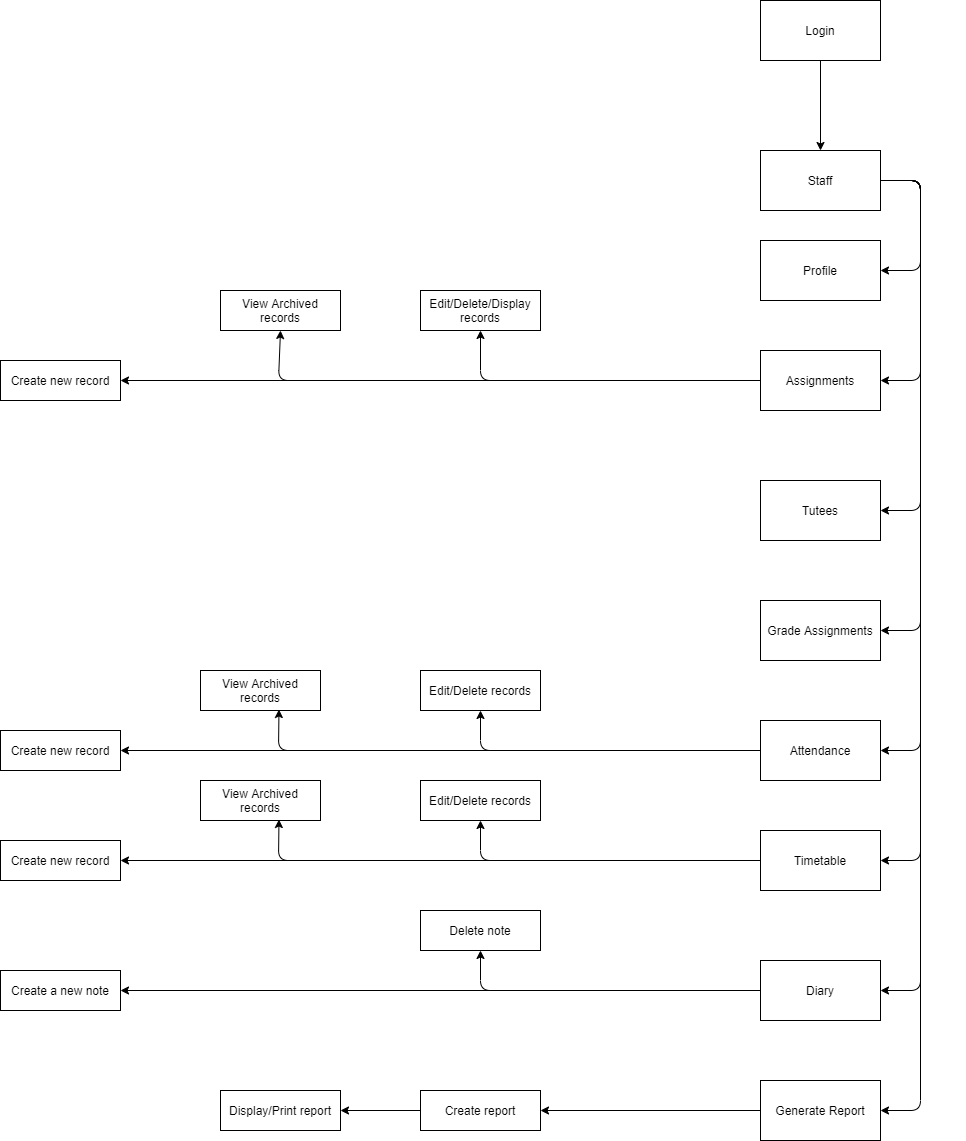


Figure 4.1..2 For Staff

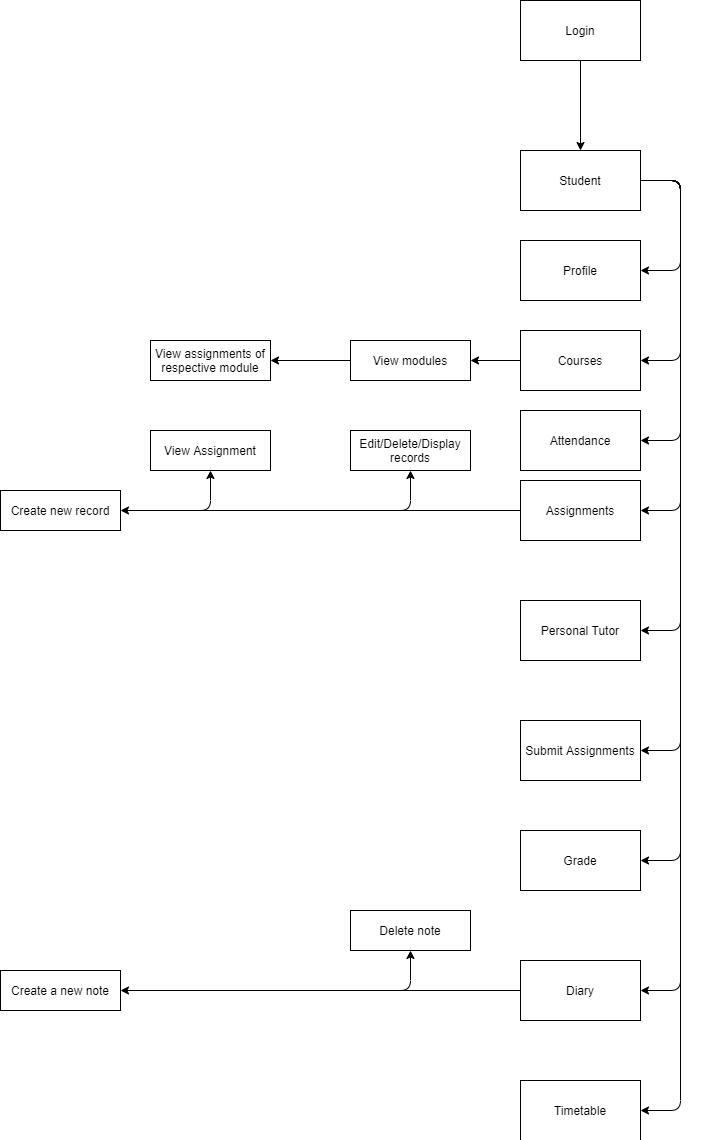


Figure 4.1..1 For Student

### **4.1.3 System Screen Mockups:**

Mockups show a better picture of how the site looks visually, and helps to plan out the best combination of elements on the screen. The following mockups show various pages that a student, staff and admin can access and has included different icons, colors, and buttons to create a draft of these pages on the site.

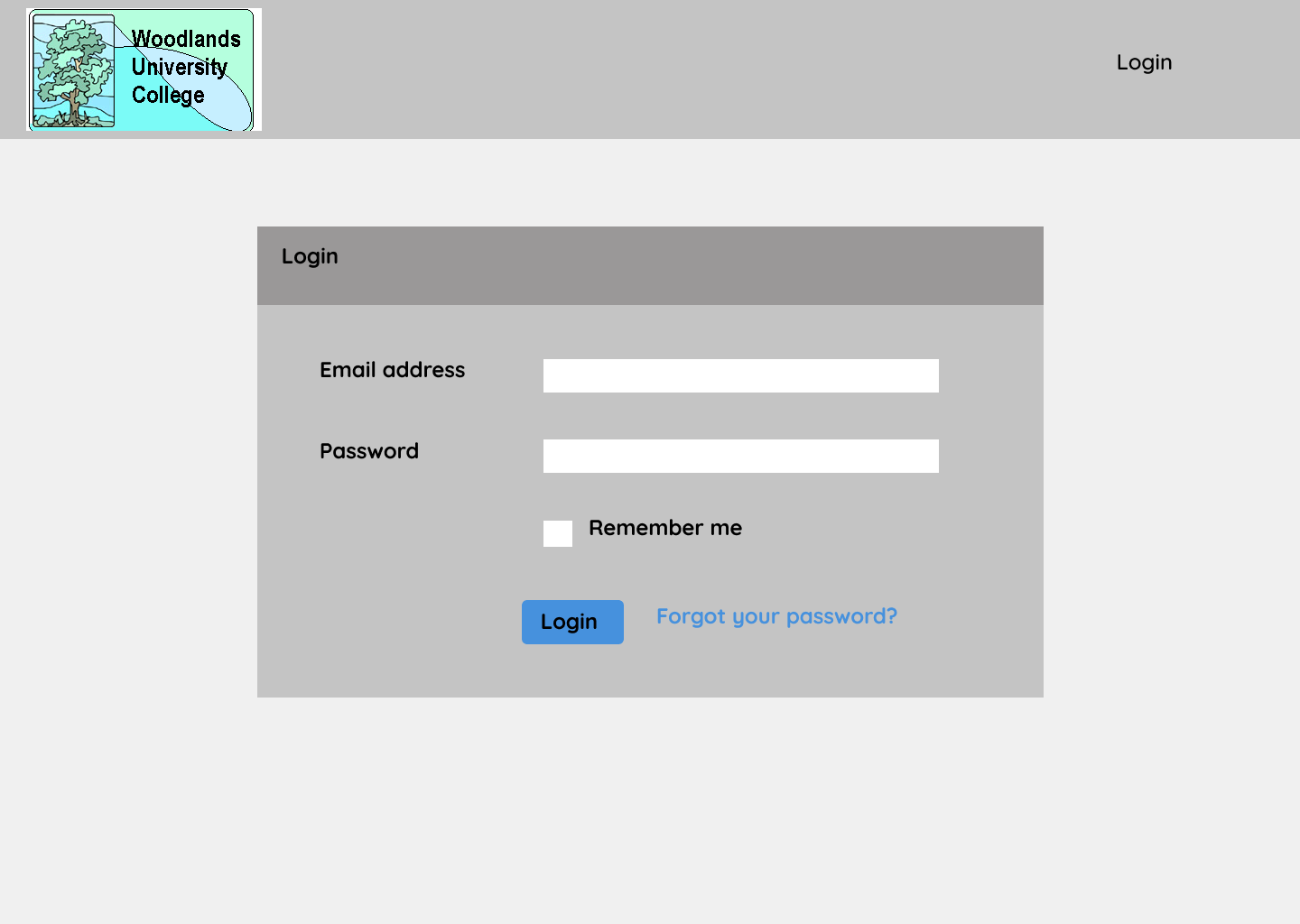
****

Figure 4.1.3.1: Log in Page

**Pages from Admin’s view:**

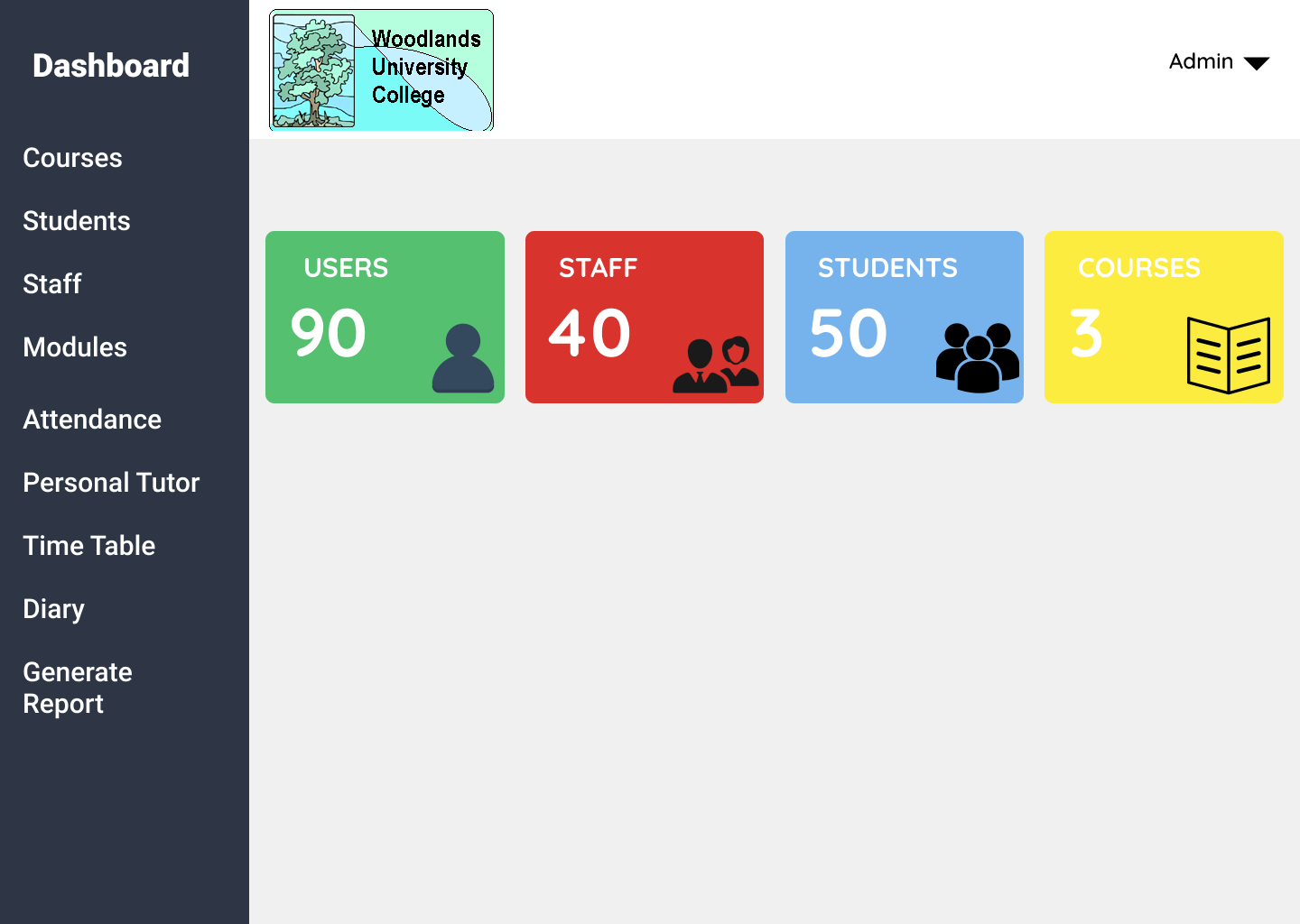


Figure 4.1.3.2: Home Page (ADMIN)

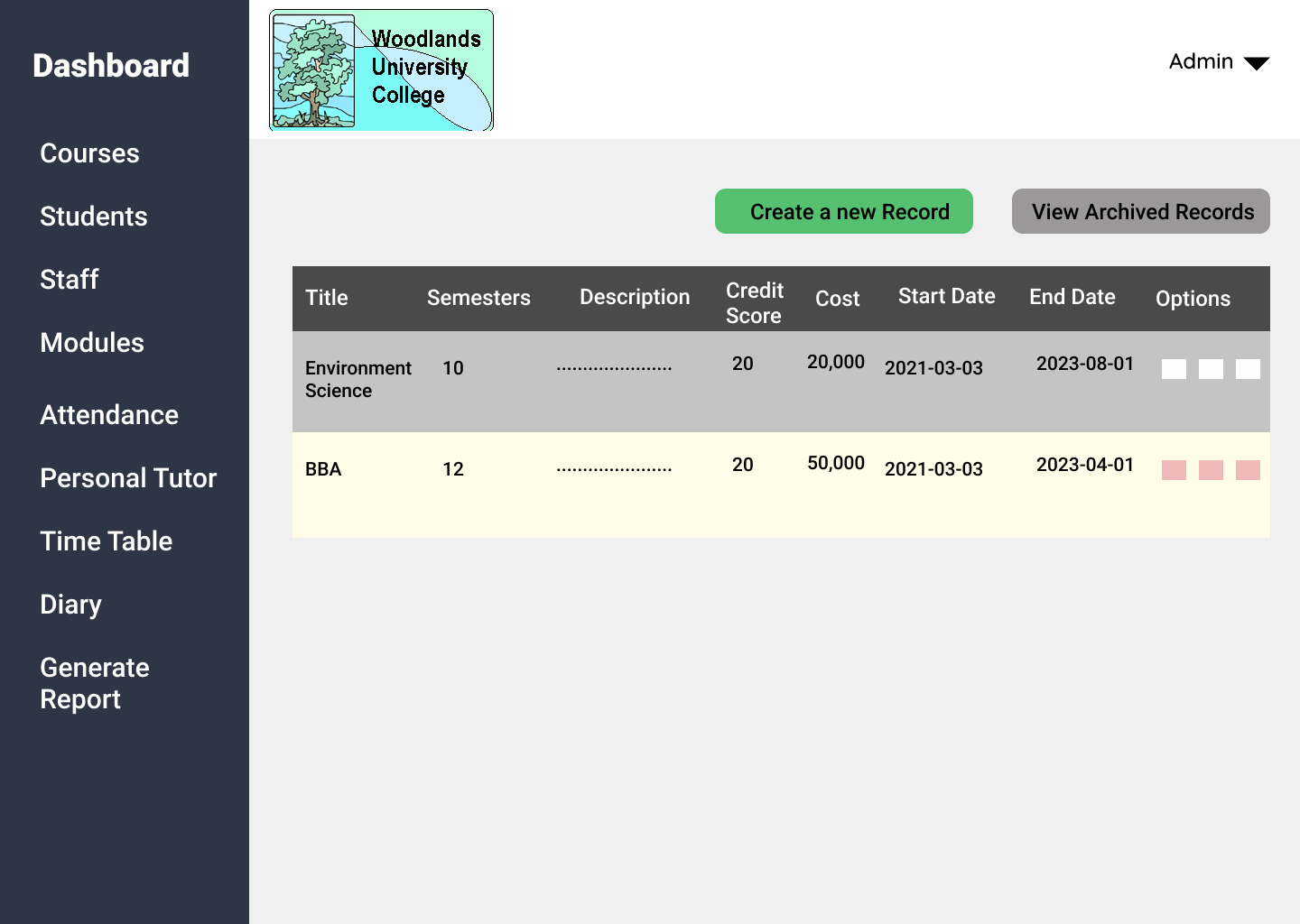


Figure 4.1.3.3: Courses Page

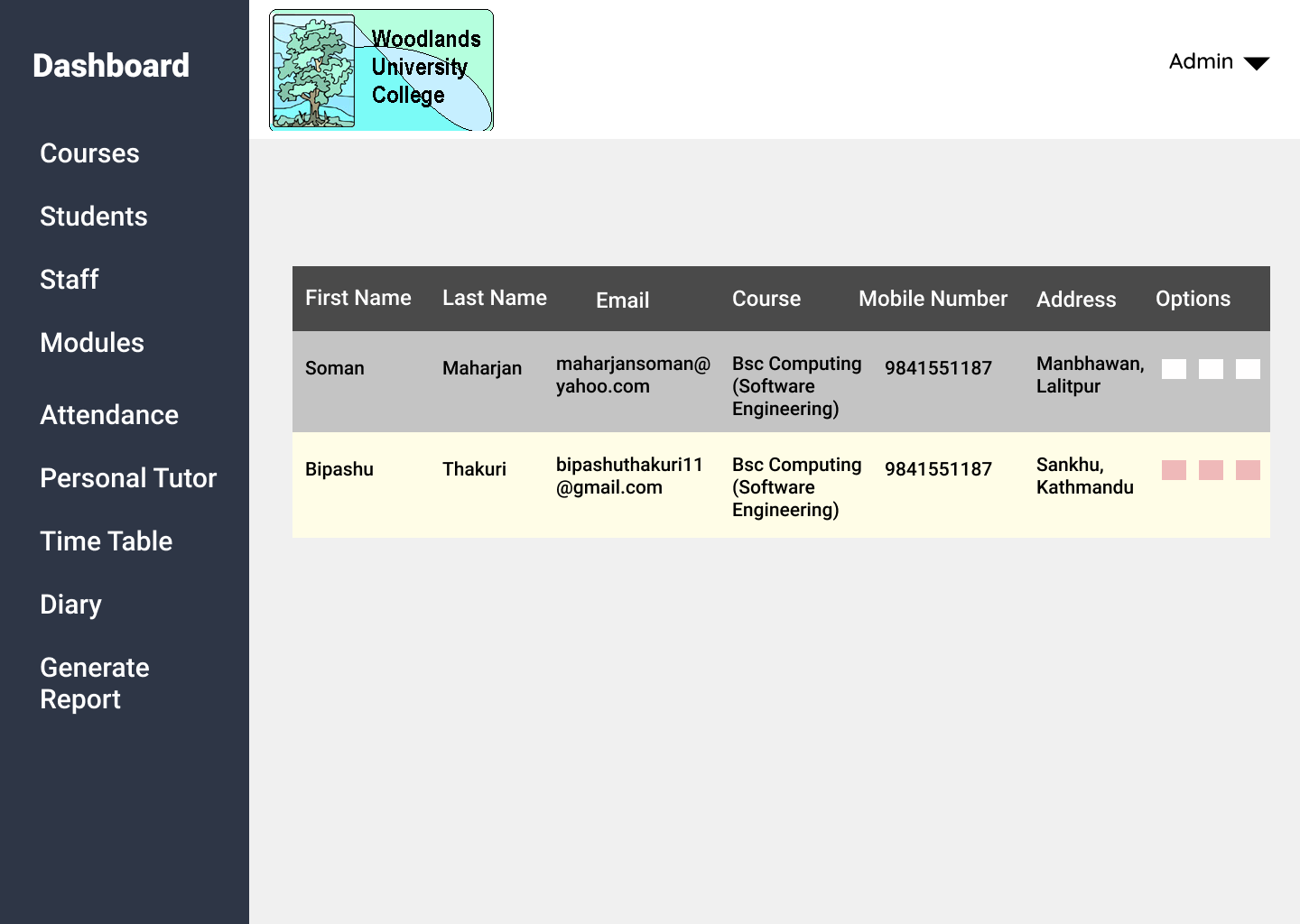


Figure 4.1.3. 4: Students Page

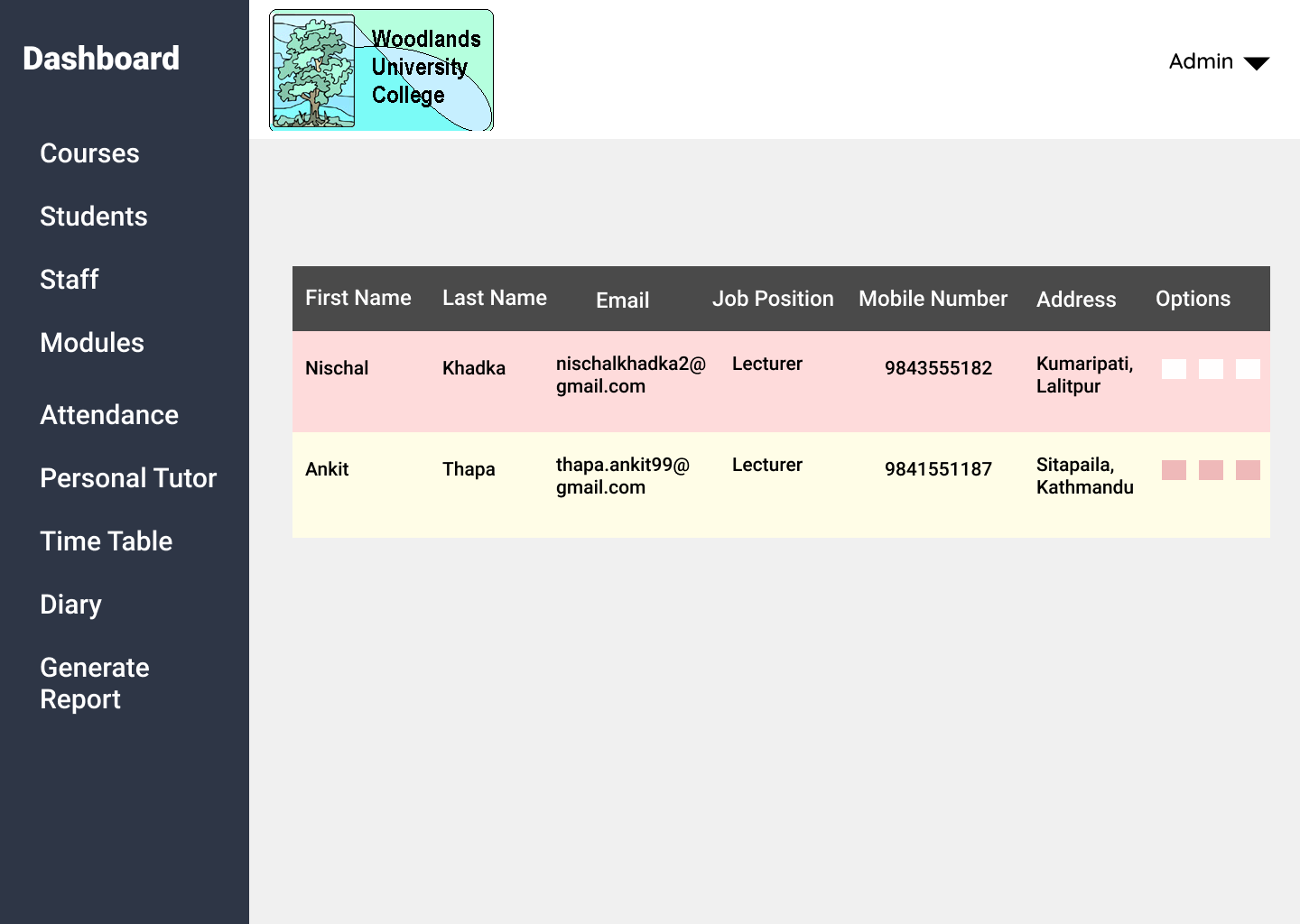


Figure 4.1.3.5: Staff Page

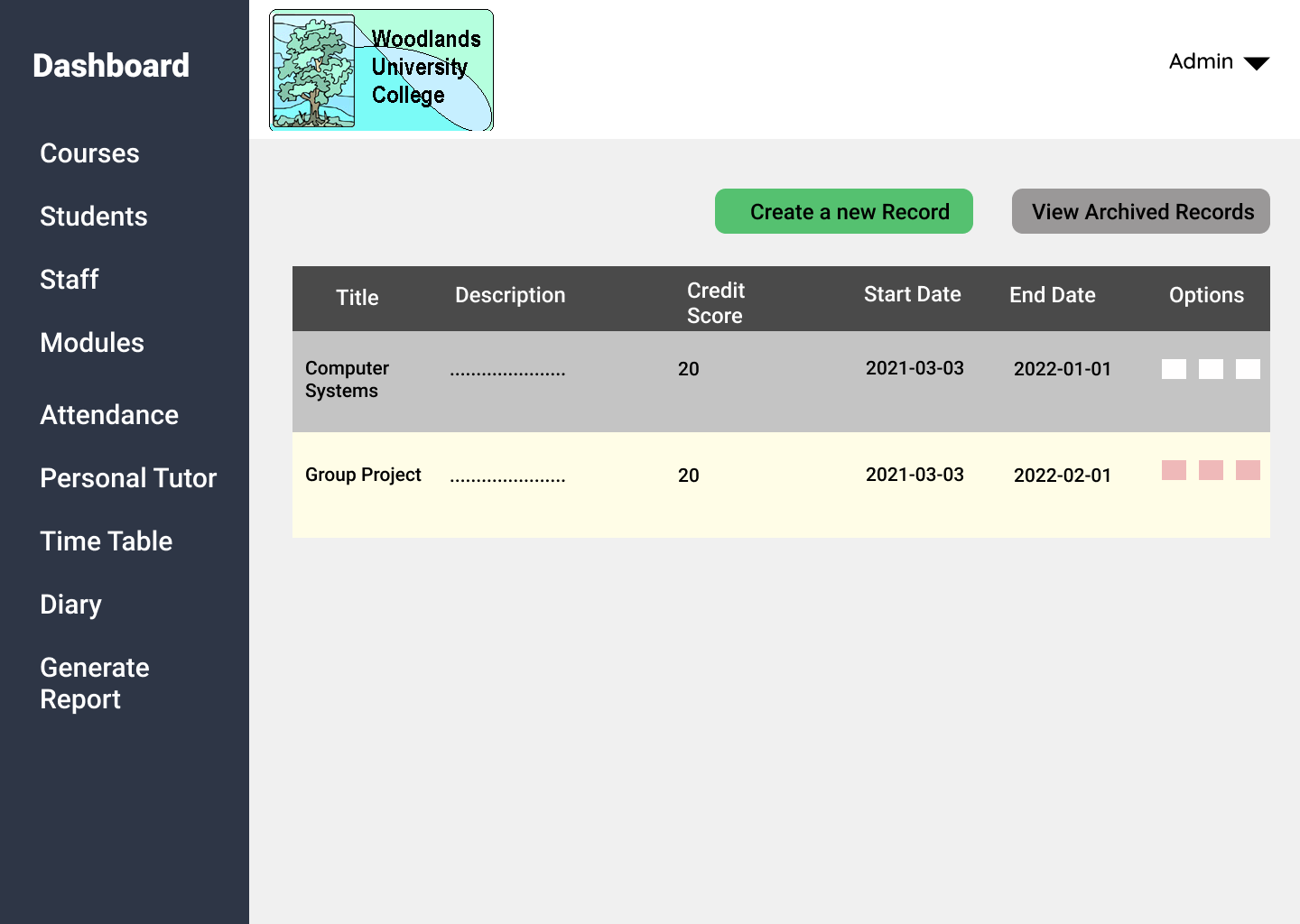


Figure 4.1.3.6: Modules Page

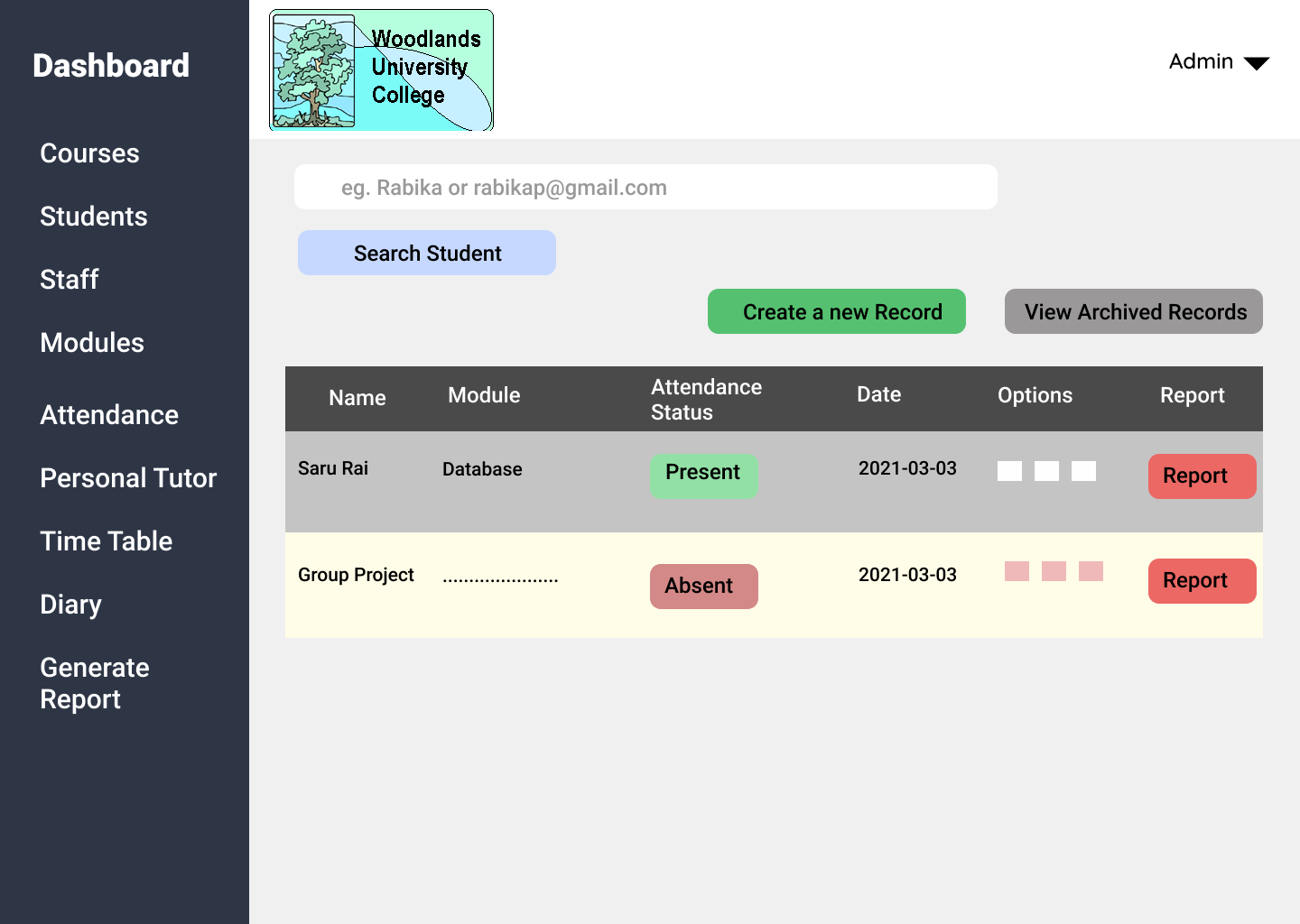


Figure 4.1.3.7: Attendance Page

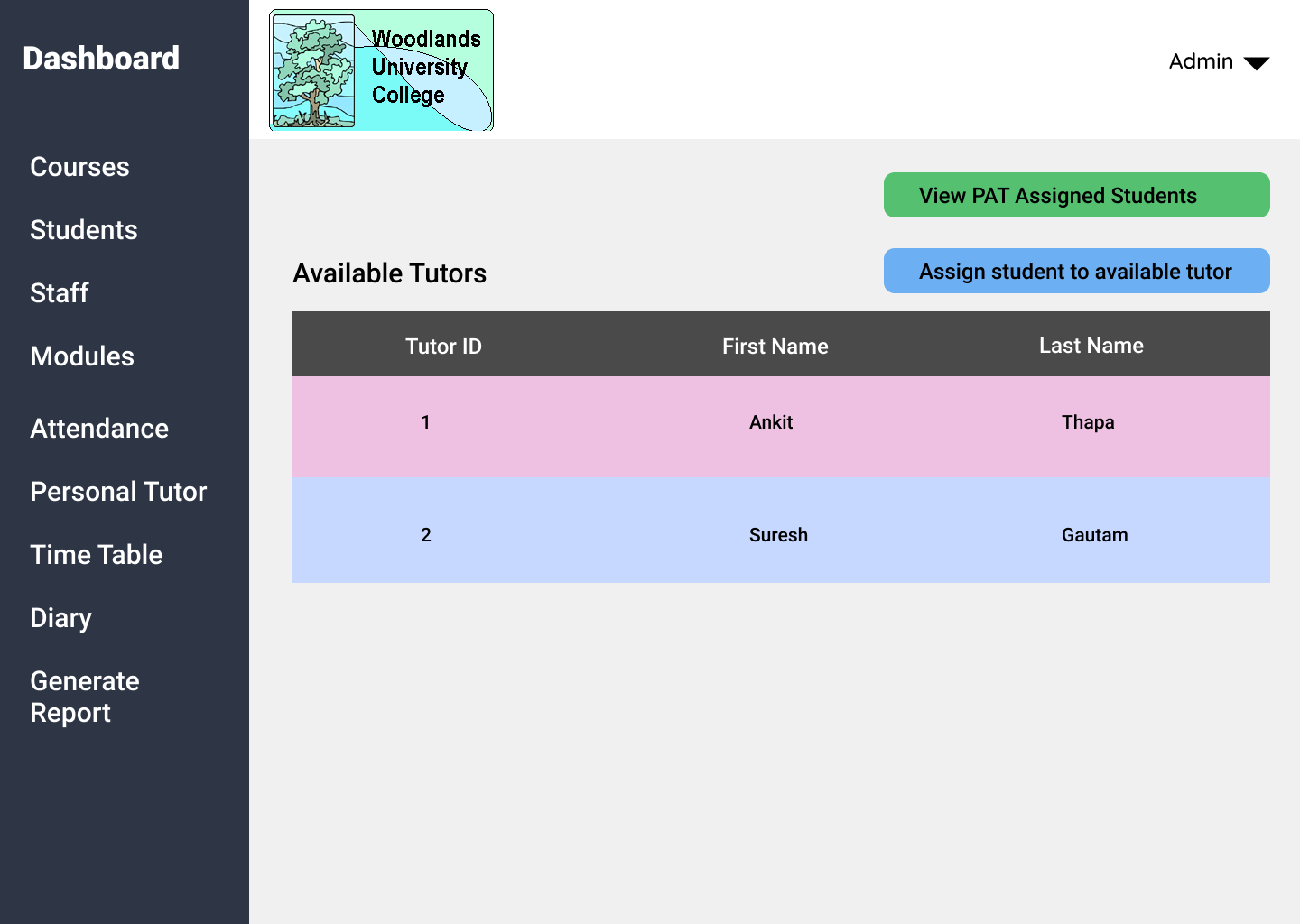


Figure 4.1.3.8: Personal Tutor Page

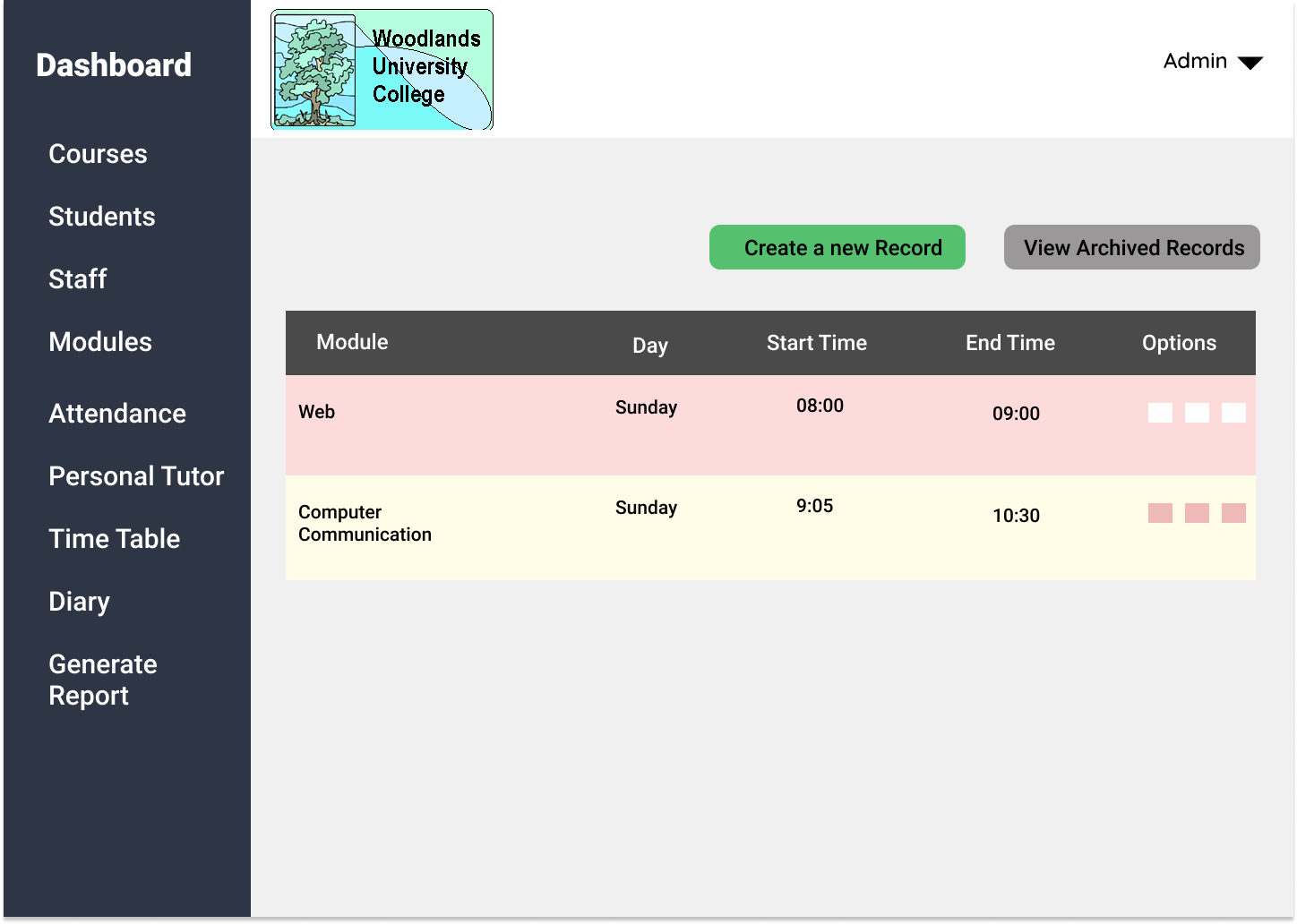


Figure 4.1.3.9: Timetable

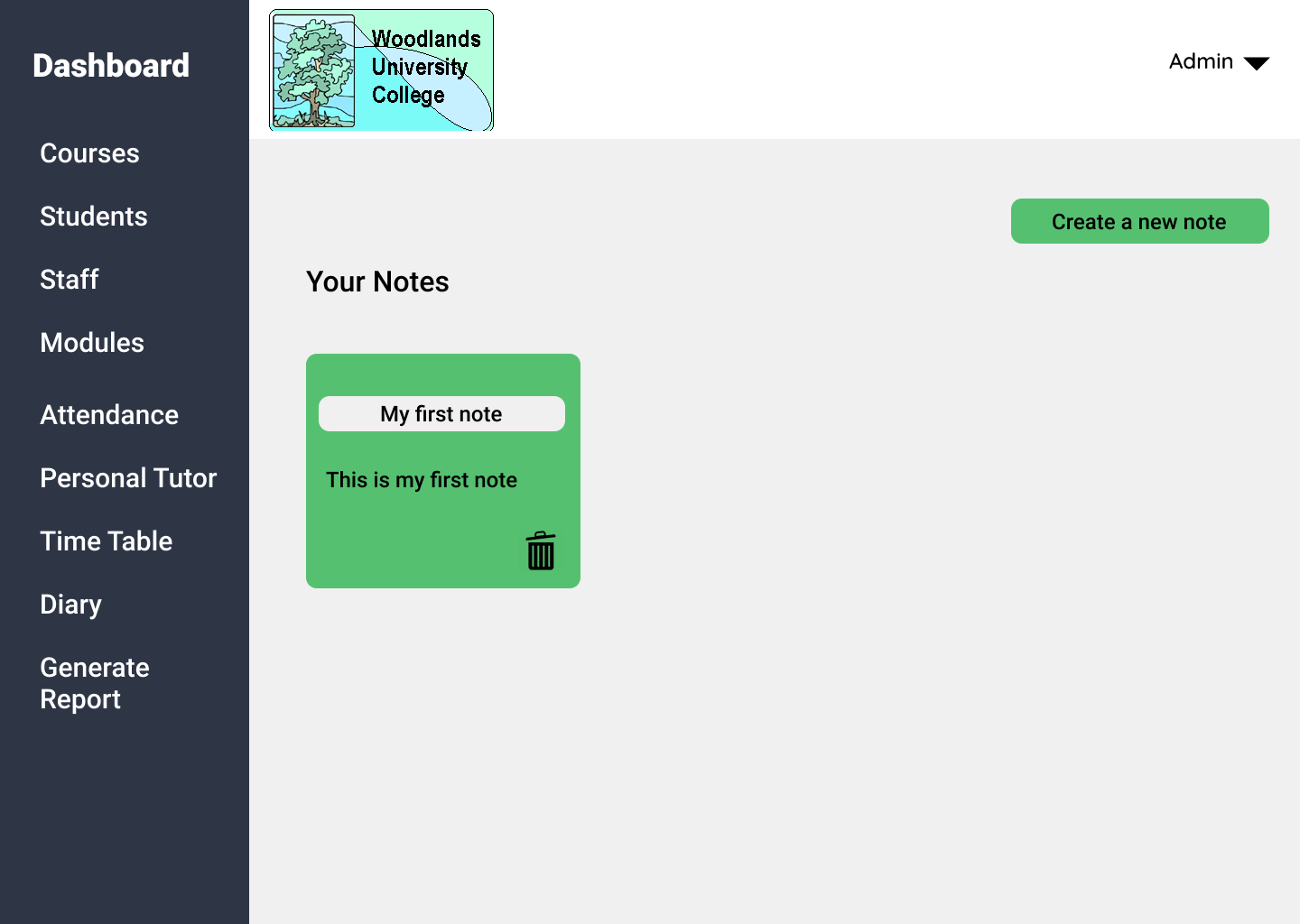


Figure 4.1.3.10: Diary

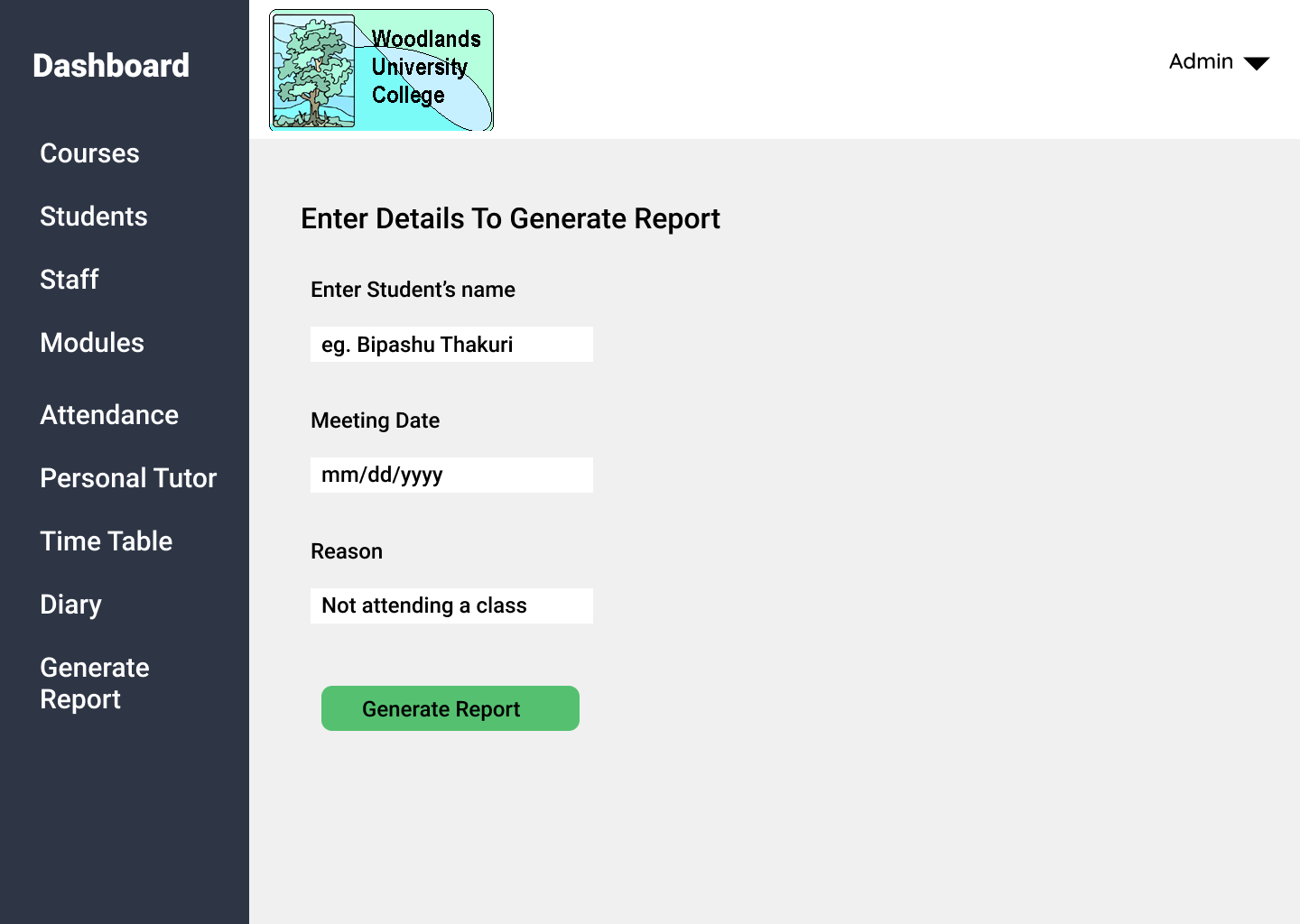


Figure 4.1.3.11: Generate Report

**Pages from Staff’s view:**

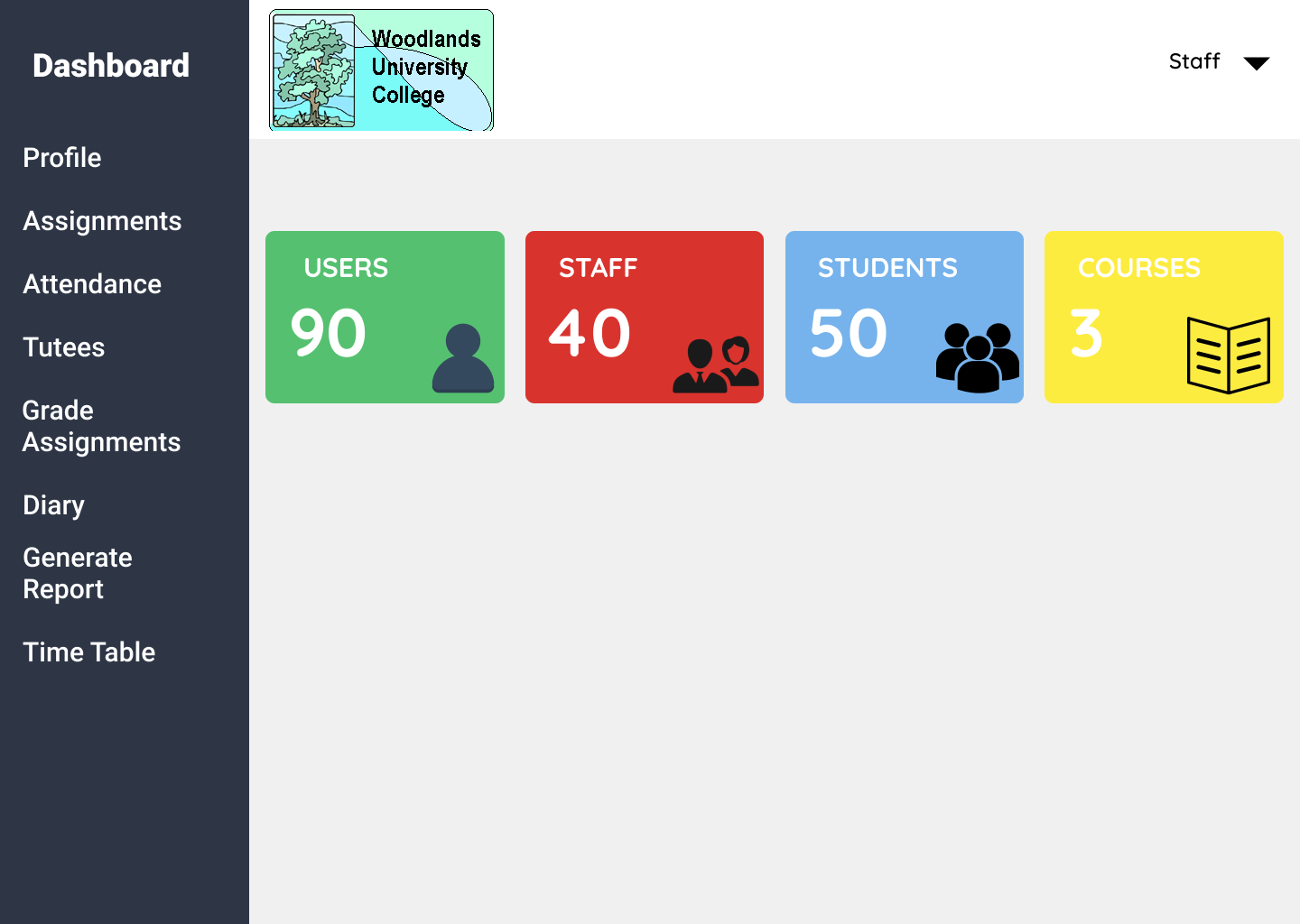


Figure 4.1.3.12: Home Page (STAFF)

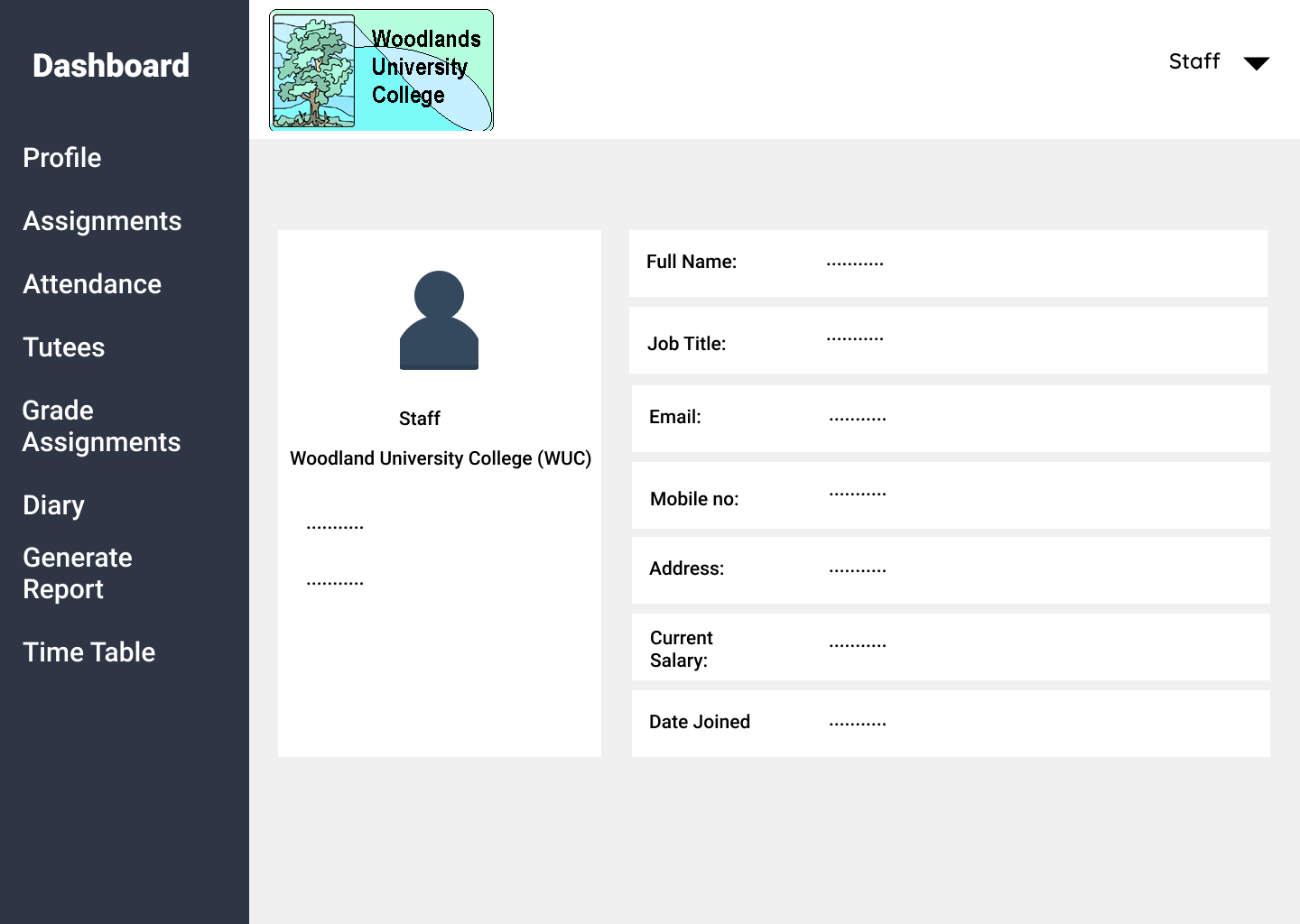


Figure 4.1.3.13: Profile Page

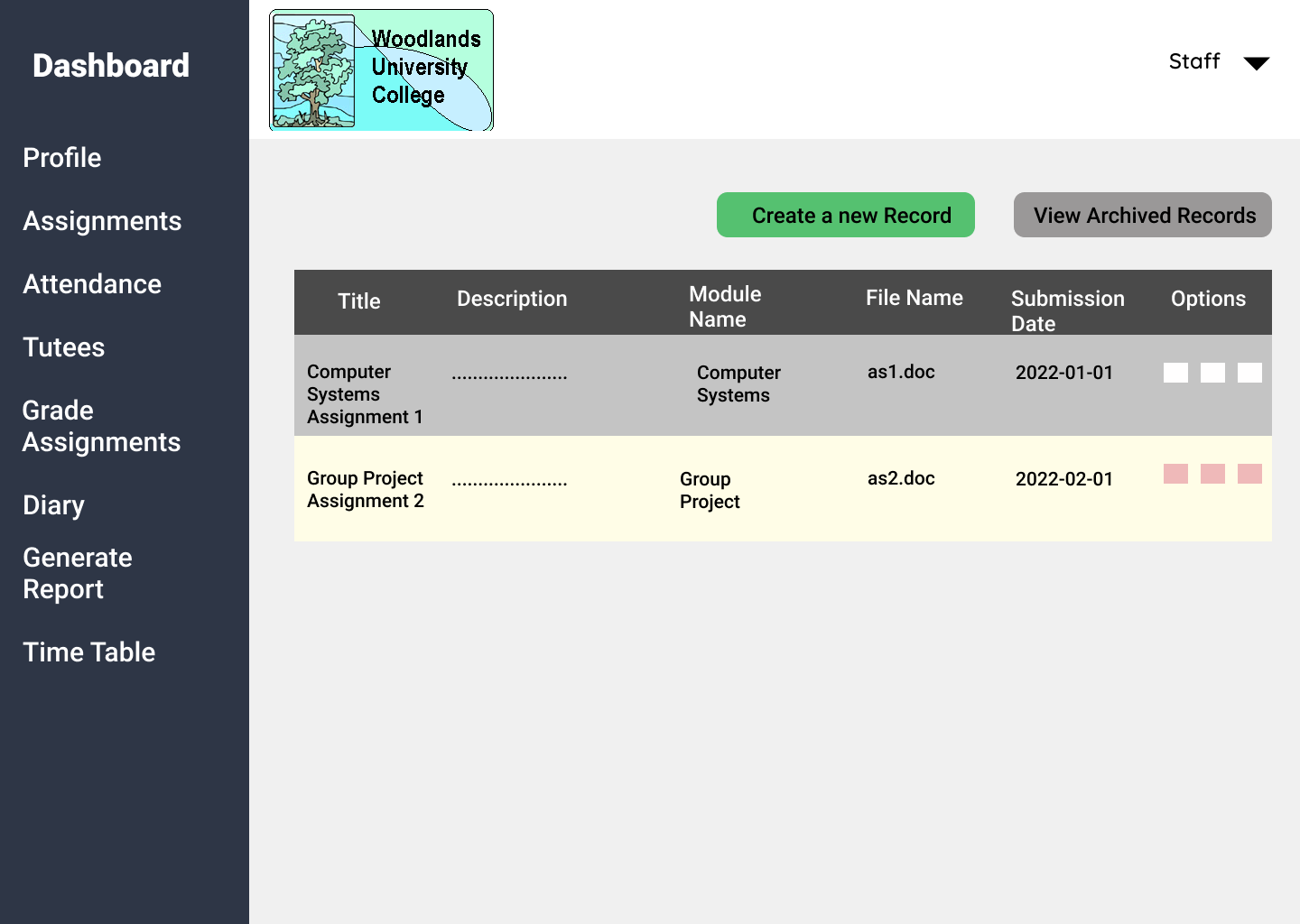


Figure 4.1.3.13: Assignments Page

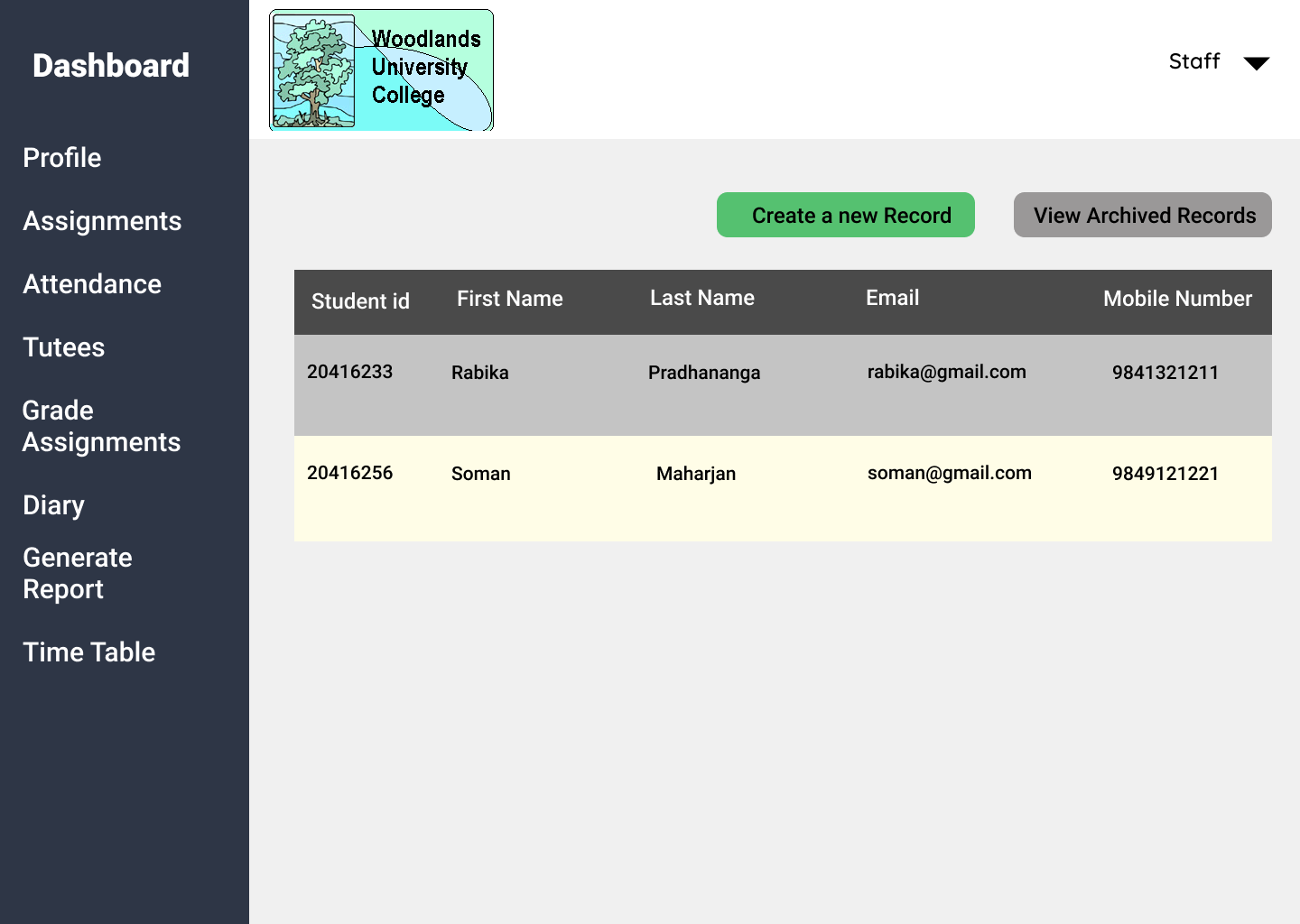


Figure 4.1.3.14: Tutees Page

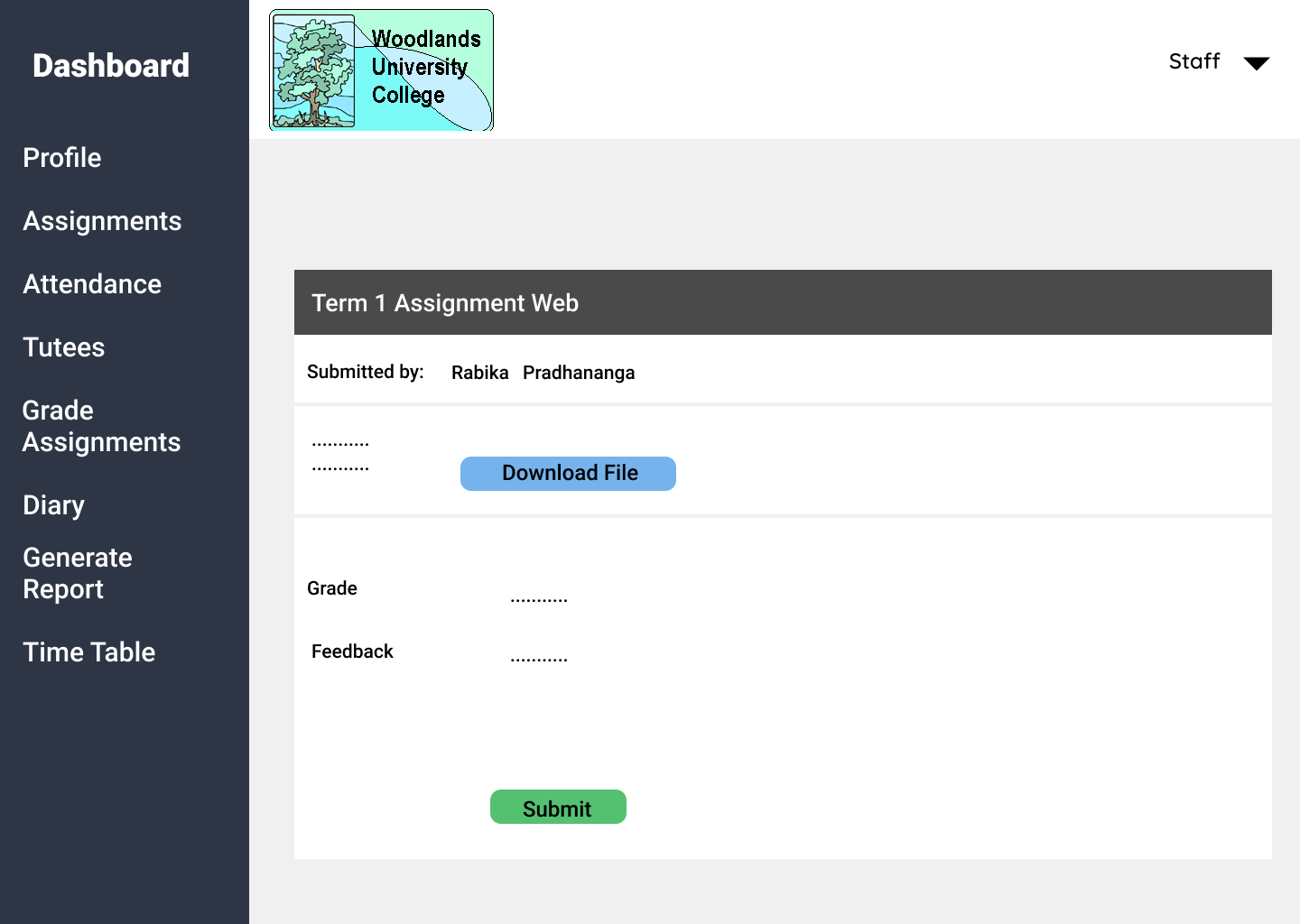


Figure 4.1.3.15: Grade Assignments Page

Attendance, Timetable, Diary, and Generate Report (4.1.3.7, 4.1.3.9, 4.1.3.10, 4.1.3.11) as seen from admin’s view to be included for staff as well.

**Pages from Student’s view:**

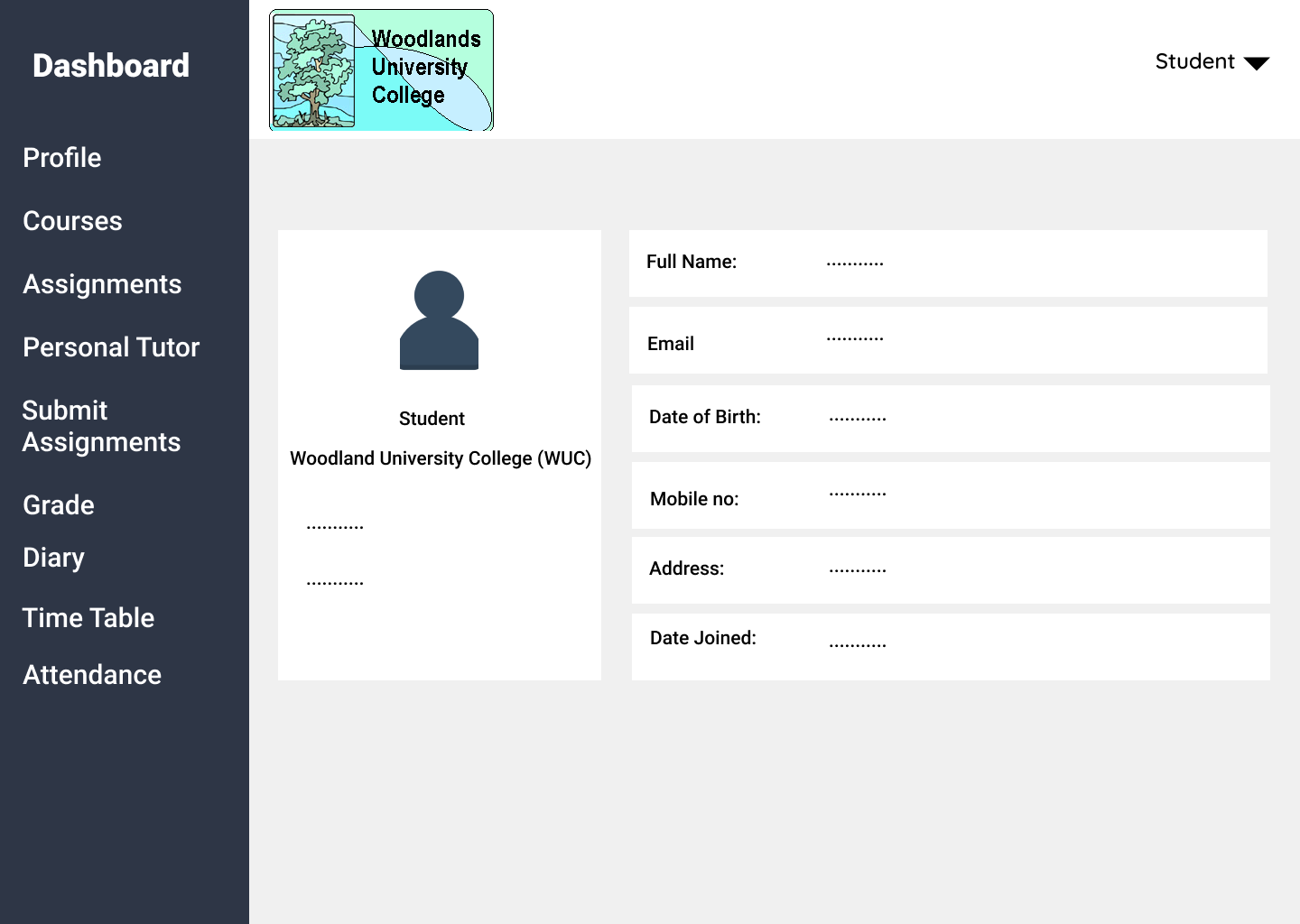


Figure 4.1.3.16: Student Profile Page

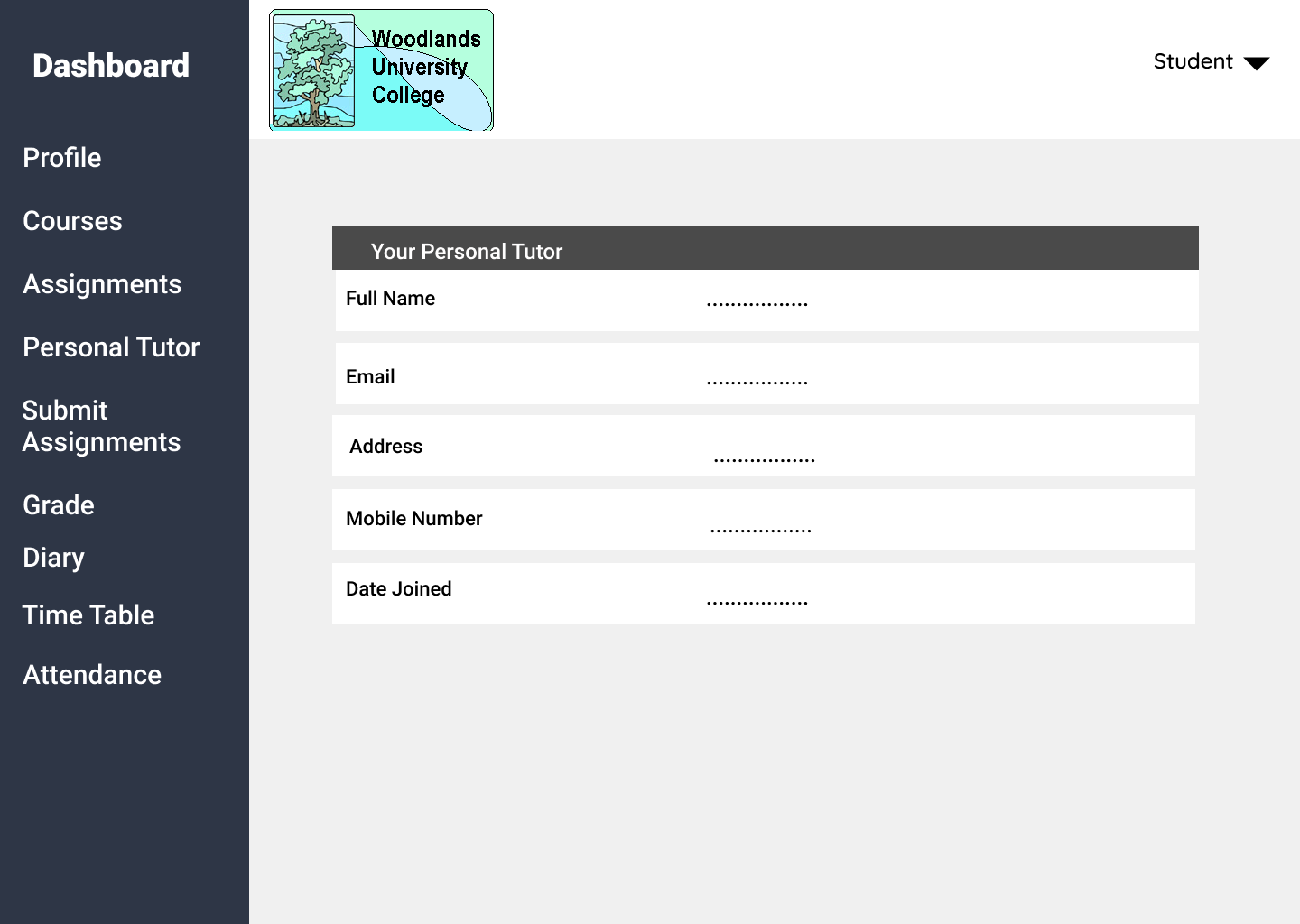


Figure 4.1.3.17: Personal Tutor Page

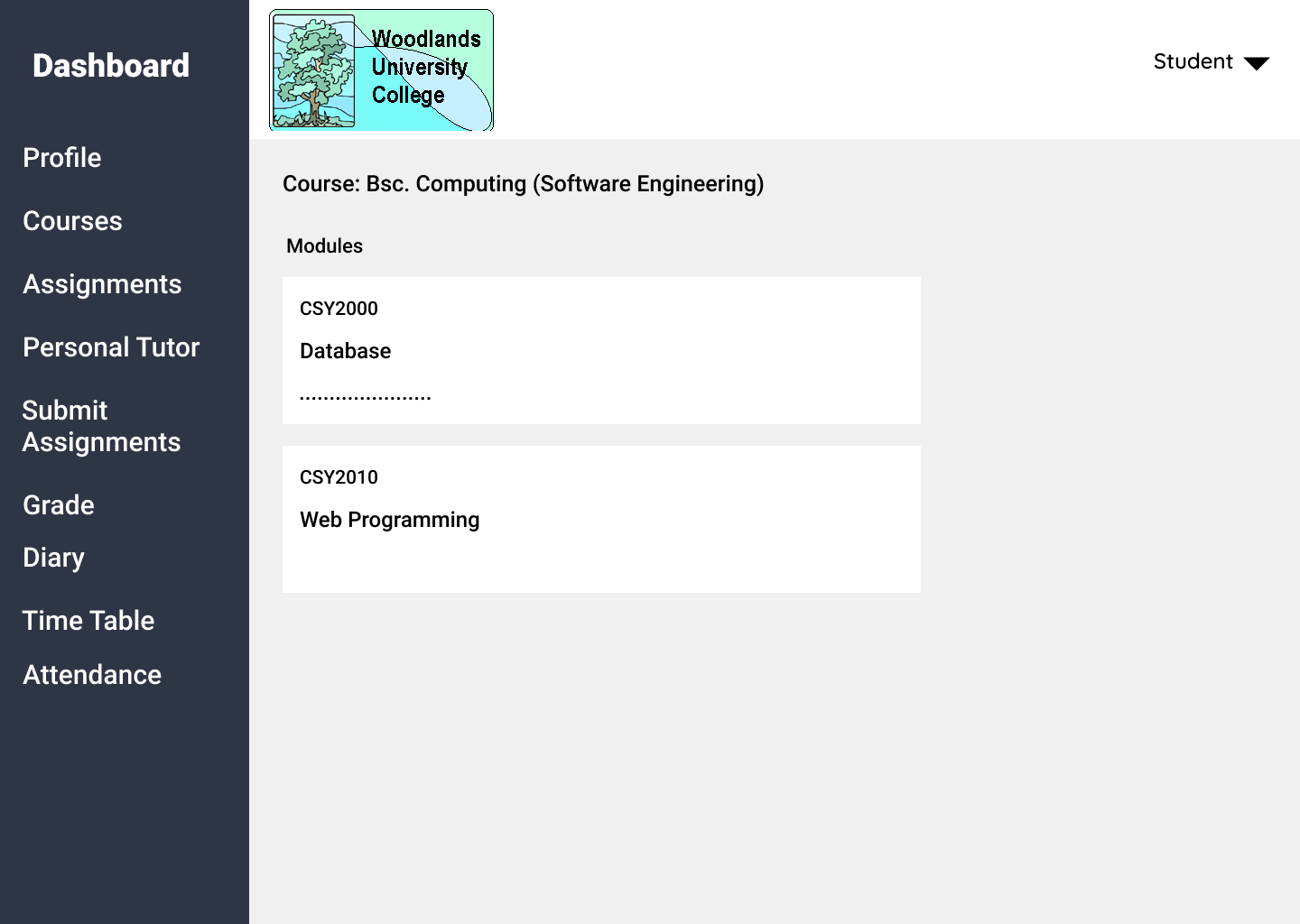
****

Figure 4.1.3.18: Courses Page

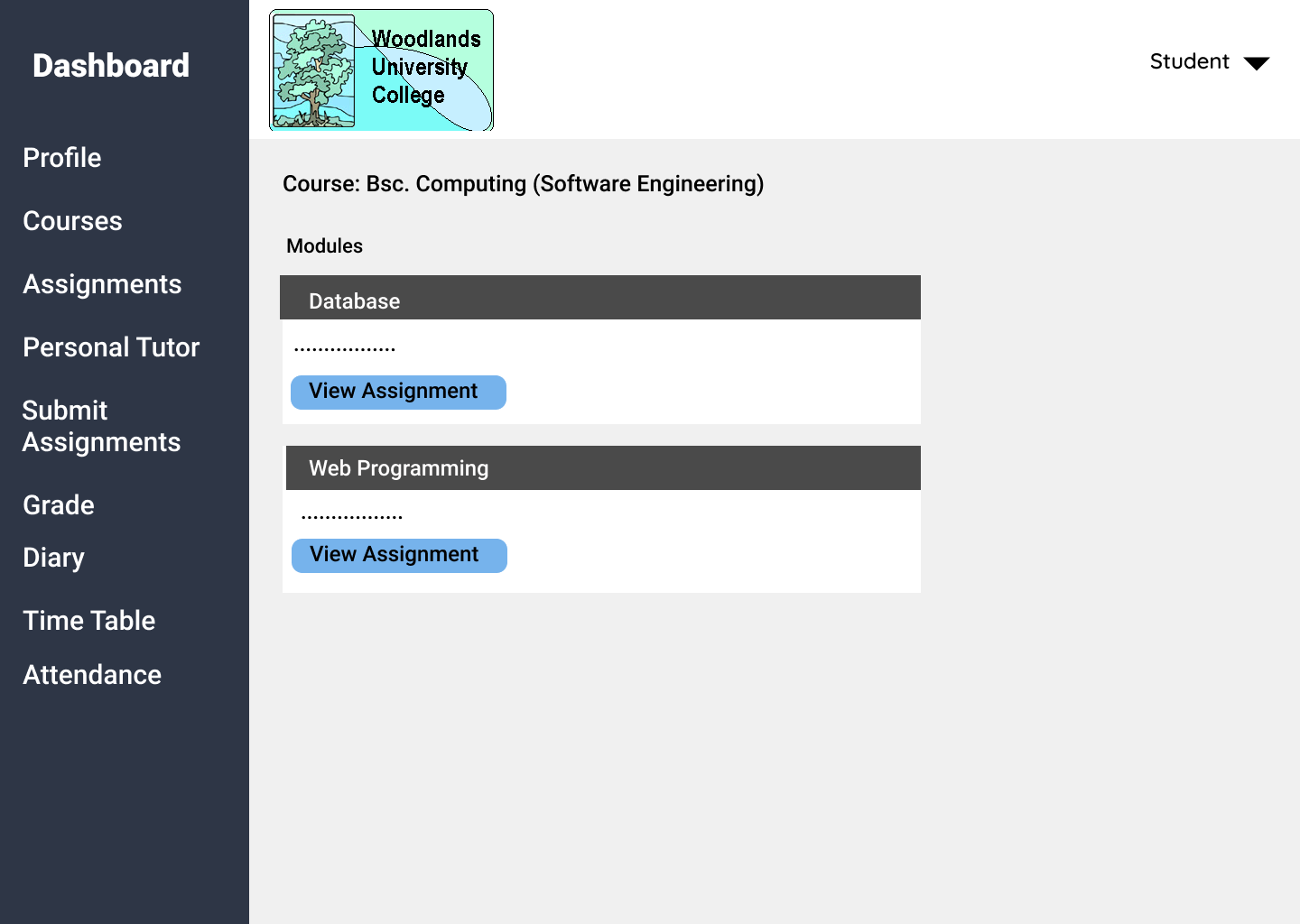
****

Figure 4.1.3.19: Assignments Page

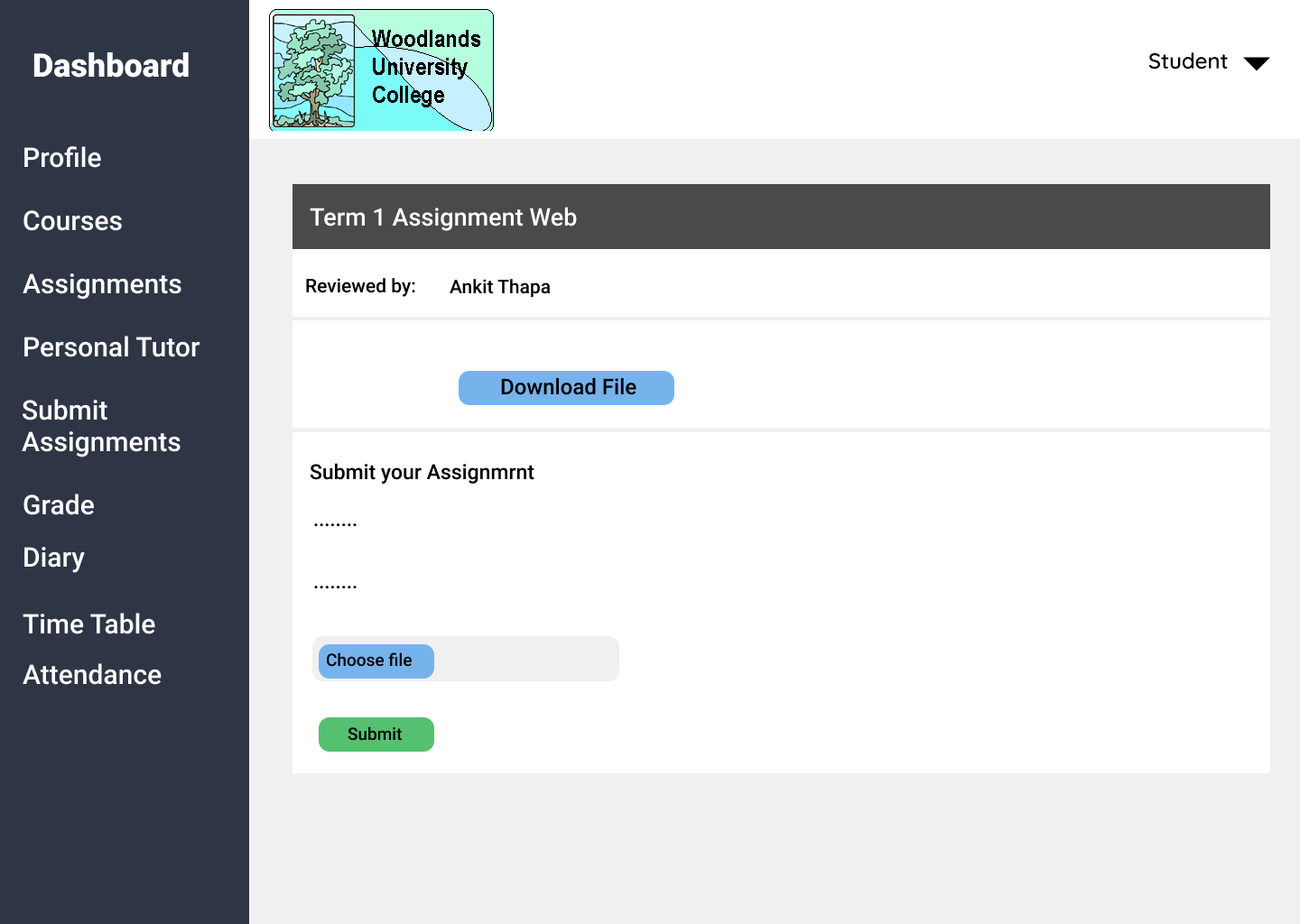
****

Figure 4.1.3.20: Submit Assignments Page

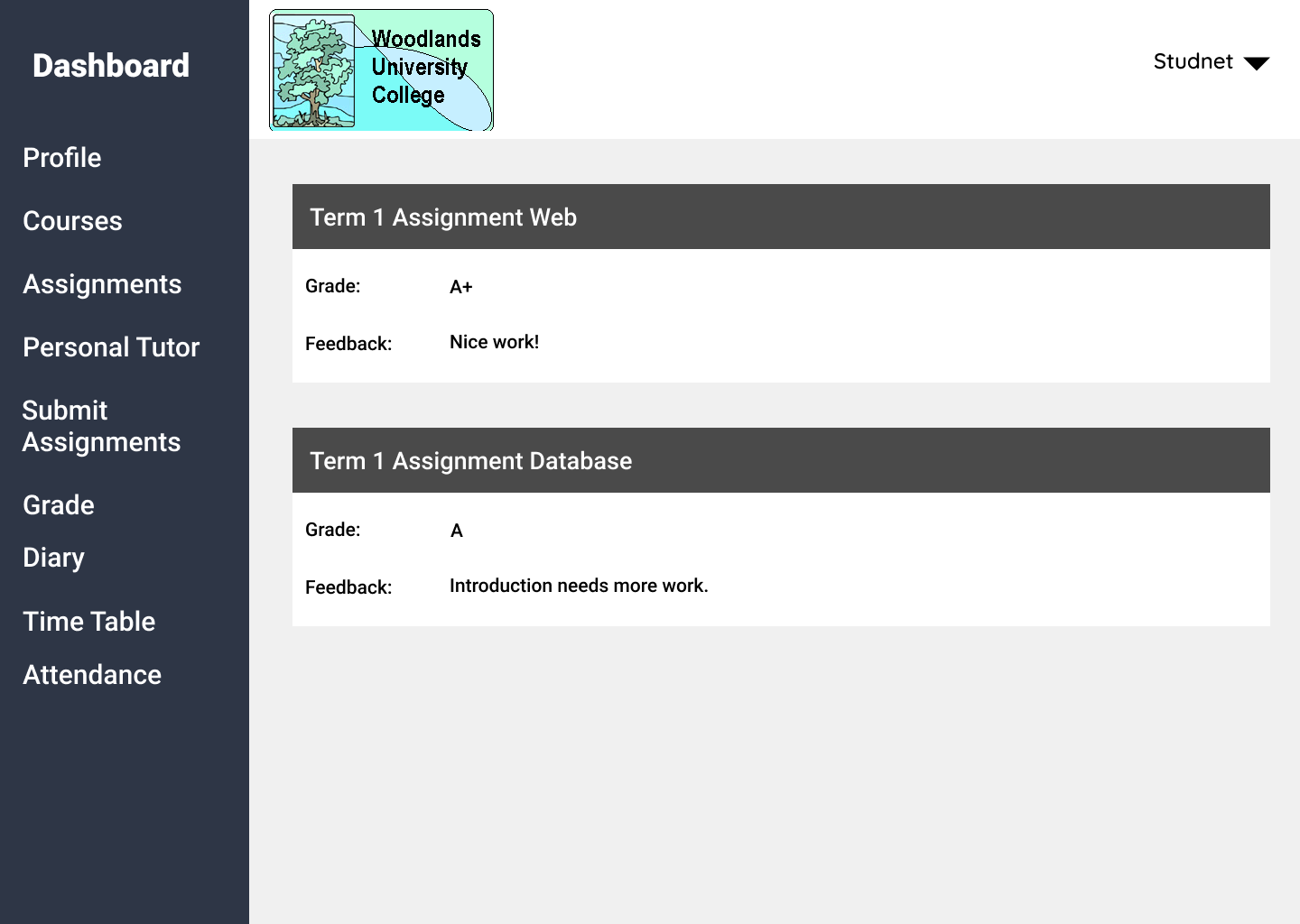
****

Figure 4.1.3.21: View Grades Page

Diary (4.1.3.10) as seen from admin’s view to be included for students as well.

### **4.1.4 System Activity Event Diagrams:**

Diagram

Description automatically generated

Figure 4.1.4.1: Add and login as new Student

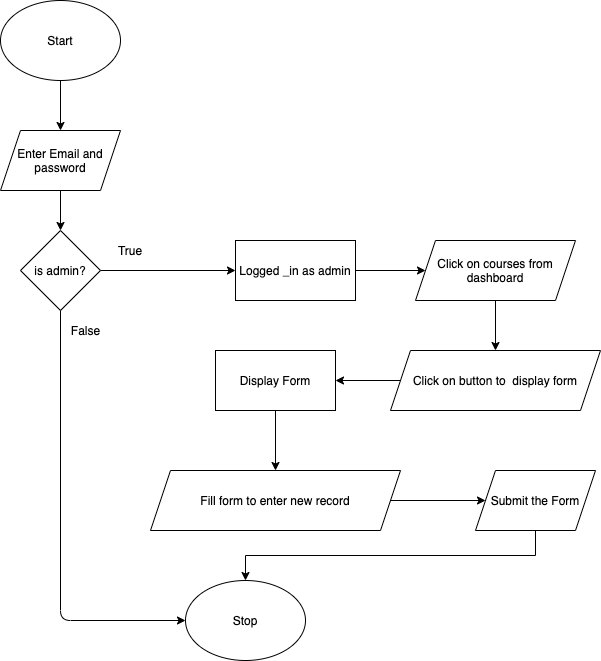


Figure 4.1.4.2: Add new course

Diagram

Description automatically generated

Figure 4.1.4.: Check attendance

Diagram

Description automatically generated

Figure 4.1.4. : Generate Report

Diagram

Description automatically generated

Figure 4.1.4.: Create assignment

Diagram

Description automatically generated

Figure 4.1.4.: Submit Assignment

Diagram

Description automatically generated

Figure 4.1.4.: Grade Assignment

Diagram

Description automatically generated

Figure 4.1.4.: View personal tutor profile

Diagram

Description automatically generated

Figure 4.1.4.: View grades

Diagram

Description automatically generated

Figure 4.1.4.: Submit assignment

Diagram

Description automatically generated

Figure 4.1.4.:View own profile

Diagram

Description automatically generated

Figure 4.1.4.: View timetable

Diagram

Description automatically generated

Figure 4.1.4.: Create new note

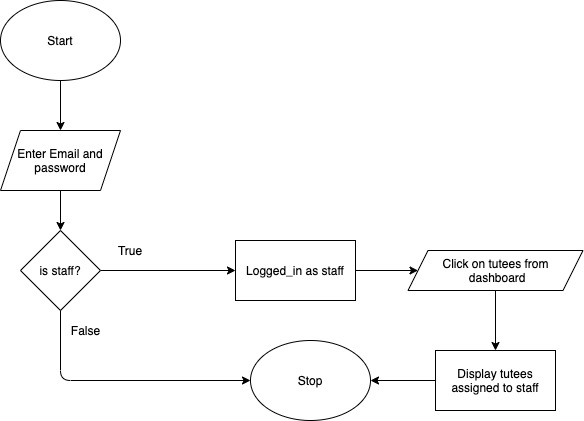


Figure 4.1.4.: View tutees

Diagram

Description automatically generated

Figure 4.1.4.15: Archive Course

Diagram

Description automatically generated

Figure 4.1.4.4.16: Assign Personal tutor

## Design Revisions:

Although the final system design is very close to the mockups that were prepared before the development process started, some design revisions have been made after consulting the client and after discussions with the team throughout the development process. Some key design changes that were made throughout the process are given below:

* Use of appropriate icons for functionalities such as editing, deleting or displaying records, to assist the user and make the site easy to use
* Dashboards added for admin, staff and student where the landing page for students is their own profile, whereas admin and staff can see information such as the number of users, courses, and students displayed along with helpful icons
* Use of uniform colors to display data instead of using different colors for every row which made it too colorful previously
* Name of the student or staff is shown in the right corner after they log in successfully, where they can choose to log out from the drop-down arrow

## Heuristic Evaluation:

Heuristic Evaluation is the way to ensure our website is user friendly. It is similar to user testing. But in user testing, testers are users but in heuristic evaluation testers are usability experts. Usability experts are professional who are skilled in determining heuristic evaluation criteria. We will use 10 usability heuristics for the heuristic evaluation.

1. Visibility of system status.

The system should keep users informed of what’s going on.

1. Match between the system and the real world.

The system should use language that is easier for targeted users to understand. This is extremely important for error messages.

1. User Control and Freedom.

Users should be able to undo and redo actions and also have the ability to move backward. This is important since people make mistakes.

1. Consistency and Standards.

Screen almonds should be consistent across the entire platform such as font style, font size and colours.

1. Error prevention.

Error should be kept to a minimum. You should try and prevent errors from happening by crafting careful design so that users cannot break system.

1. Recognition rather than recall.

Minimize the users load by making information from one part of the website relevant to the other part of the website.

1. Flexibility and efficiency of use.

Shortcuts allows experts to speed up the experience while being unseen to the range of users.

1. Aesthetic and minimalist design.

Irrelevant information should not be included in the design. Design should be limited to must have elements only. Like landing page should be very simplistic and easy to navigate.

9. Helps users recognize, diagnose and recover from errors.

Error messages should use plain language without codes. It should indicate the problem and suggest solution.

10. Help and documentation.

There should be additional documentation to provide the users with additional information.

|  |  |
| --- | --- |
| Heuristics | Result |
| Visibility of system status. | ✓ |
| Match between the system and the real world. | ✓ |
| User Control and Freedom. | ✓ |
| Consistency and Standards. | ✓ |
| Error prevention. | ✓ |
| Recognition rather than recall. | ✓ |
| Flexibility and efficiency of use. | ✓ |
| Aesthetic and minimalist design. | ✓ |
| Helps users recognize, diagnose and recover from errors. | ✓ |
| Help and documentation. | ✓ |

# 5 System Build and Technical Notes:

## Technically Difficult Code Sections:

**Storing many to many relation data**



Course and module in the application has an many to many relations. In order to store the data, the database is created in such a way that it contains composite table (course\_module) between tables “courses” and “modules”. So, while storing data for course or module, the relations need to be entered in the course\_module table as well.

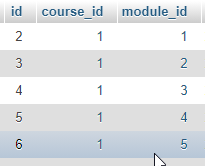


Fig. Course with course\_id 1 having relation with multiple modules

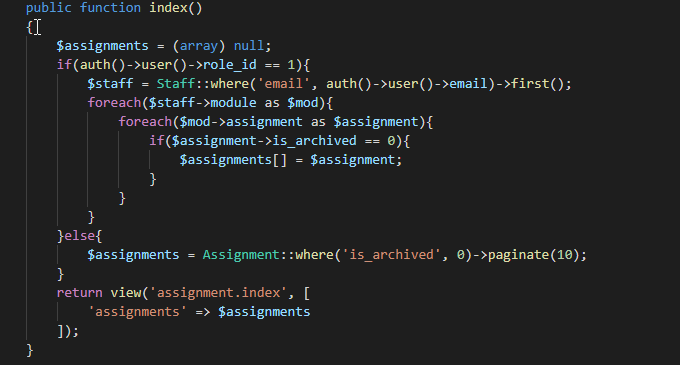
If user selects one course and assigns multiple modules then each module needs to be added with the same course id. This could be solved using foreach but instead of repeating the same code for each module, this code uses “sync” method.

First of all, the user input is validated, then the course data is saved. After that the module variable stores all the modules that are selected by the user.

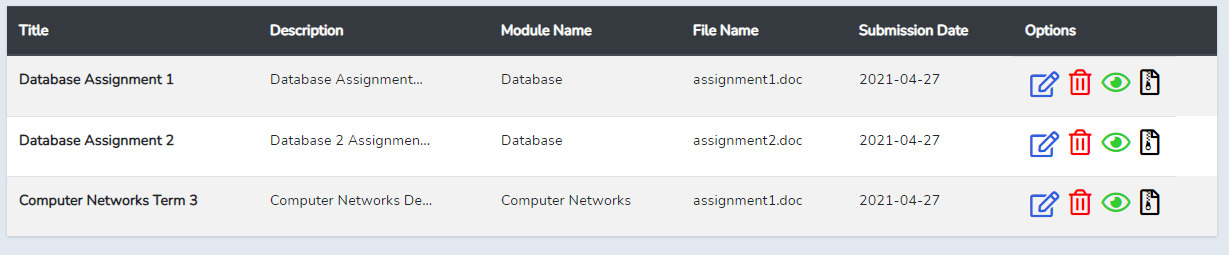
$course->module()->sync($module);

The above code stores the course id and module id of all the modules that are selected by the user. And finally, the page is redirected to “/course” with a success message.

**Display Assignments**



The above code is for Assignment page which looks like:



It returns array of assignments based on the user account type.

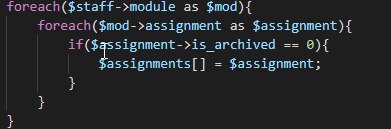
For admin:

It returns assignments page with all the assignments that are not archived.

For staff:



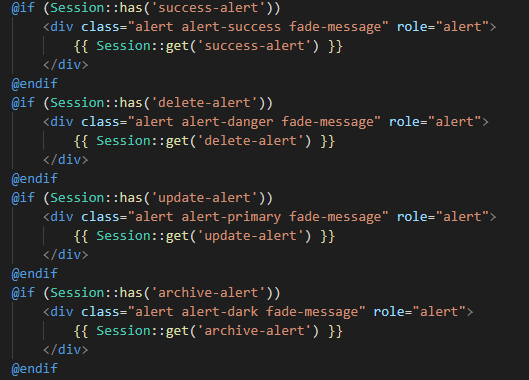
It gets the staff object where the email to equal to the email that is currently logged in.



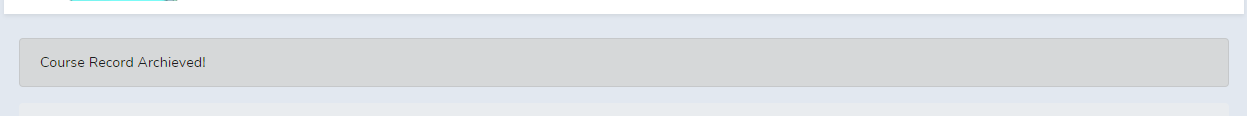
$staff->module gets modules that has relation with $staff. Then, foreach module we get the assignment with $mod->assignment. Similarly, foreach assignment we check if it is archived or not. If it is not archived, we store it in $assignments array.

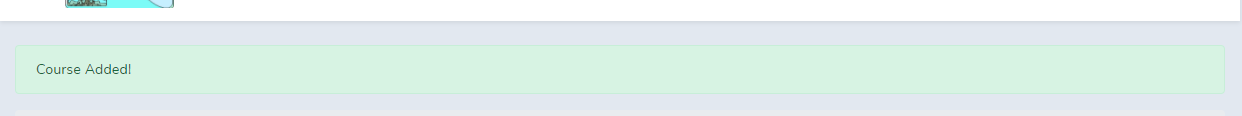
The final $assignment variable has all the unarchived assignments from the modules that the staff teaches.

**Action Alerts**

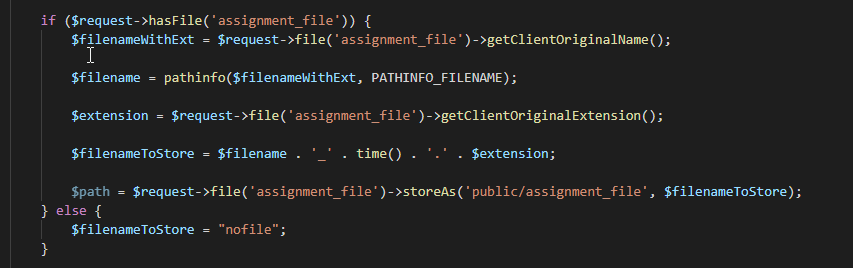


This section of the code is used to show alert when the page loads. Session, stores different type of messages in variables like “success-alert”, “delete-alert”, “update-alert”, “archive-alert” based on the action taken by the user. So, if it has variable is not empty and has value, then alerts like below gets displayed to the user. The alerts also use “fade-message” class which make the alerts fade into background, hiding it from the user.





**File Storing**



Staff account has a feature to upload file when creating a new assignment.

The first checks if the user submitted form request has file. If it has then it takes the original name of the file with getClientOriginalName() and stores it in $filenameWithExt variable. The variable is then used to get filename with pathinfo() function. Another variable $extension stores the extension part of the file.

$filenameToStore combines the filename and extension with current time to create a unique filename so that multiple files would never have the same name.

And finally, the file is stored using storeAs method to “public/assignment\_file” directory in the project with filename as $filenameToStore value.

If the user submitted form does not have file, then a string “nofile” is stored in the database.

**Modules taken by student**

Since, student does not have any relation with the module. Modules which the student is currently taking cannot be directly returned. The below functions are used to return modules that the student is taking, in order to display timetables of each module to the student.

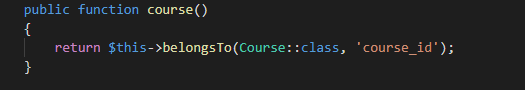


Fig. course function in Student model

If a student object is used to call the course function ($student->course) then a course object is returned such that the course id is equal to the course\_id in student object.

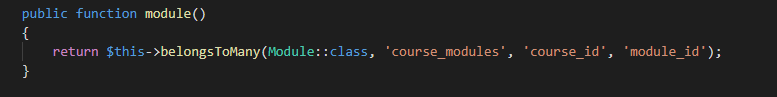


Fig. module function in Course Model

Like the course function in Student Model, the module function also works the same way. If a course object is used to call the module function, then it returns all the modules where the foreign keys are equal to the primary key in course object.

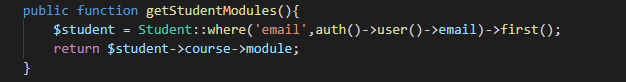


Fig. getStudentModules function in TimeTable Model

In the getStudentModules function which is used to get all the modules taken by student, both the course and module function from Student Model and Course Module is used.

First line is storing the current logged in student object to $student variable.

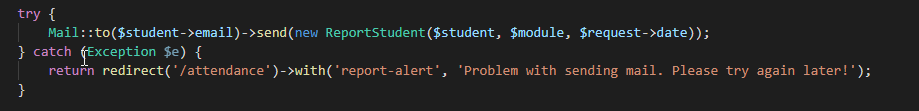
The student object is used to get the course ($student->course) which he/she is taking then the course is used to get all modules (course->module) that belongs to the course. The getStudentModules function is used in Timetable functionality to get the timetables of the modules and display it to the student.

**Email**

Staffs are able to report student if their attendance is low using “Report” button in Attendances section.



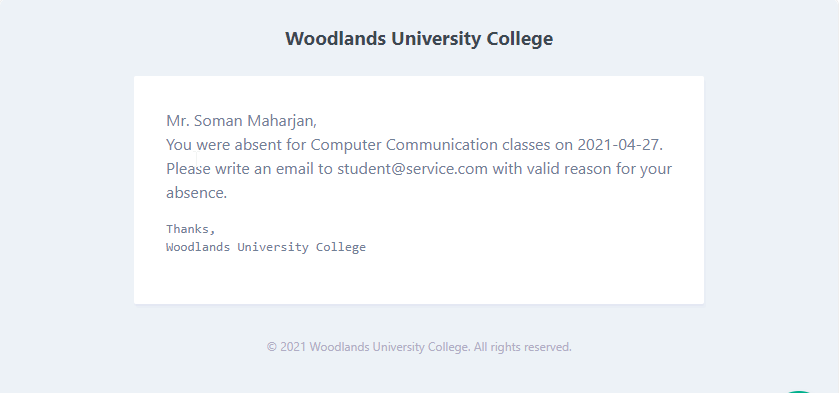
Whenever they are reported, they get an email regarding their absence with the date, module and other details.



The Mail facade is used to send direct email from college to student’s mail. It uses Report Student class which extends Mailable with a markdown template.

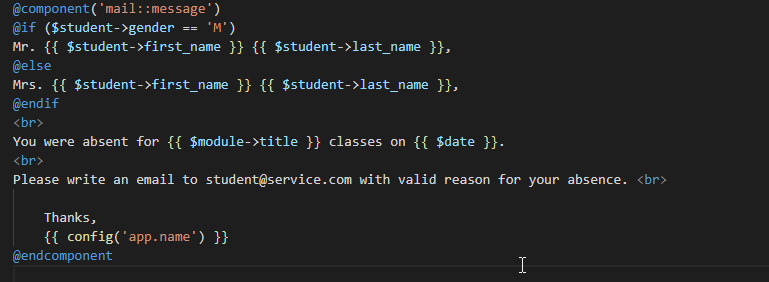
The code tries to send mail, but if it fails the exception is caught and the page is redirected to “/attendance” with a message.

The mail sent to student looks like:



It contains all the information like module name, class date and name of student.

All this html is rendered using the component below:



## On-going Testing Methodology:

Testing is essential for the success of a software. Efficient testing will result in the smooth running of the program with minimum bugs. The method of testing used is black-box testing, in which a test case is checked and compared with the predicted outcome.

## 5.3 Final System Interface Displays:

**Pages from Admin’s view:**

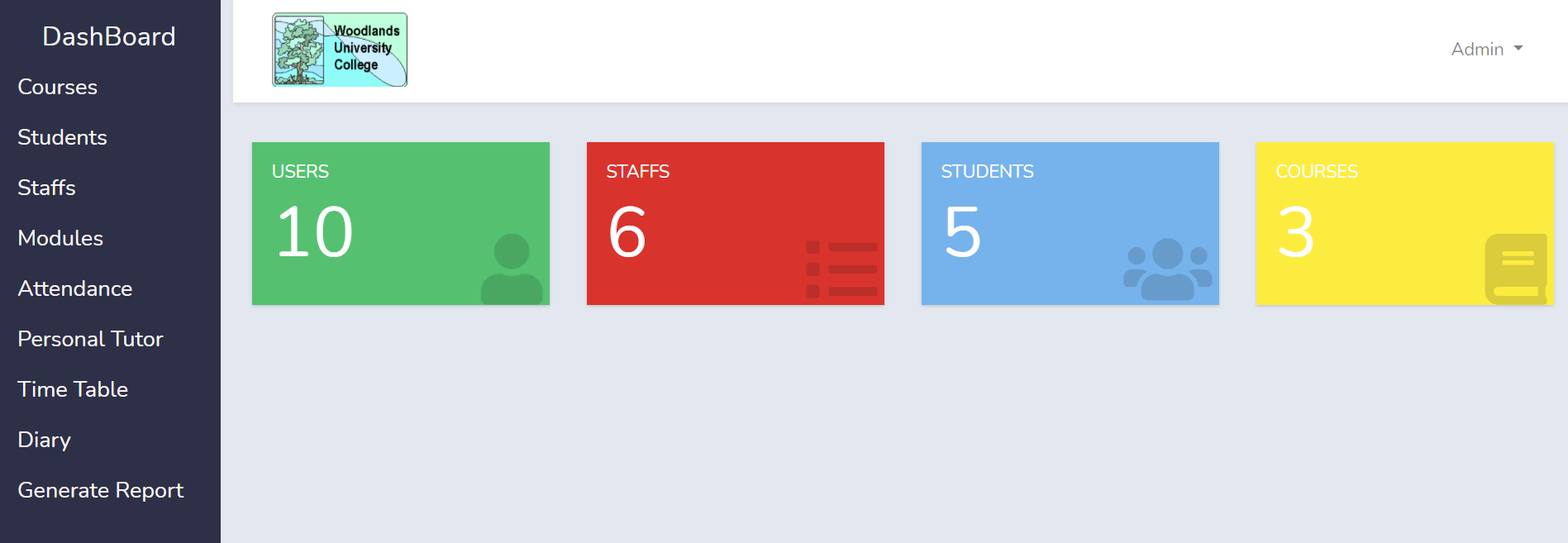


Figure 5.3.1: Admin Landing Page

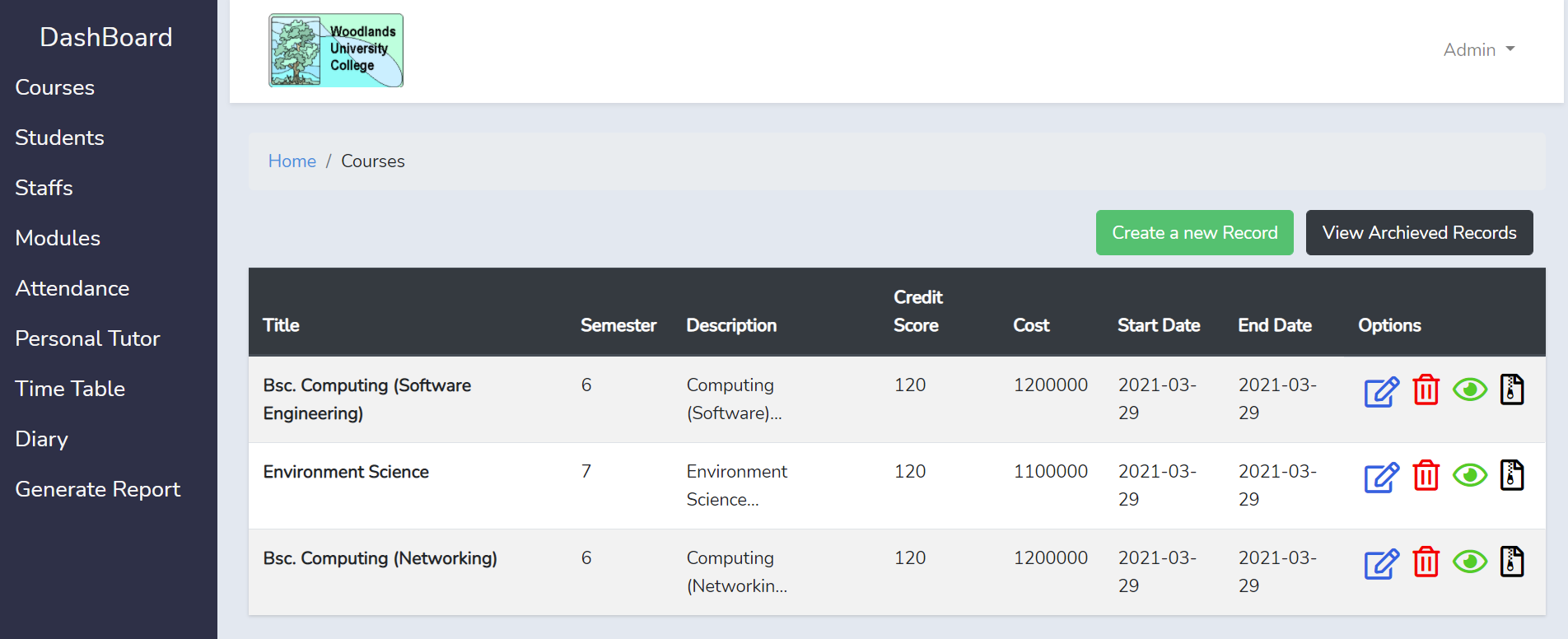


Figure 5.3.2: Courses Page

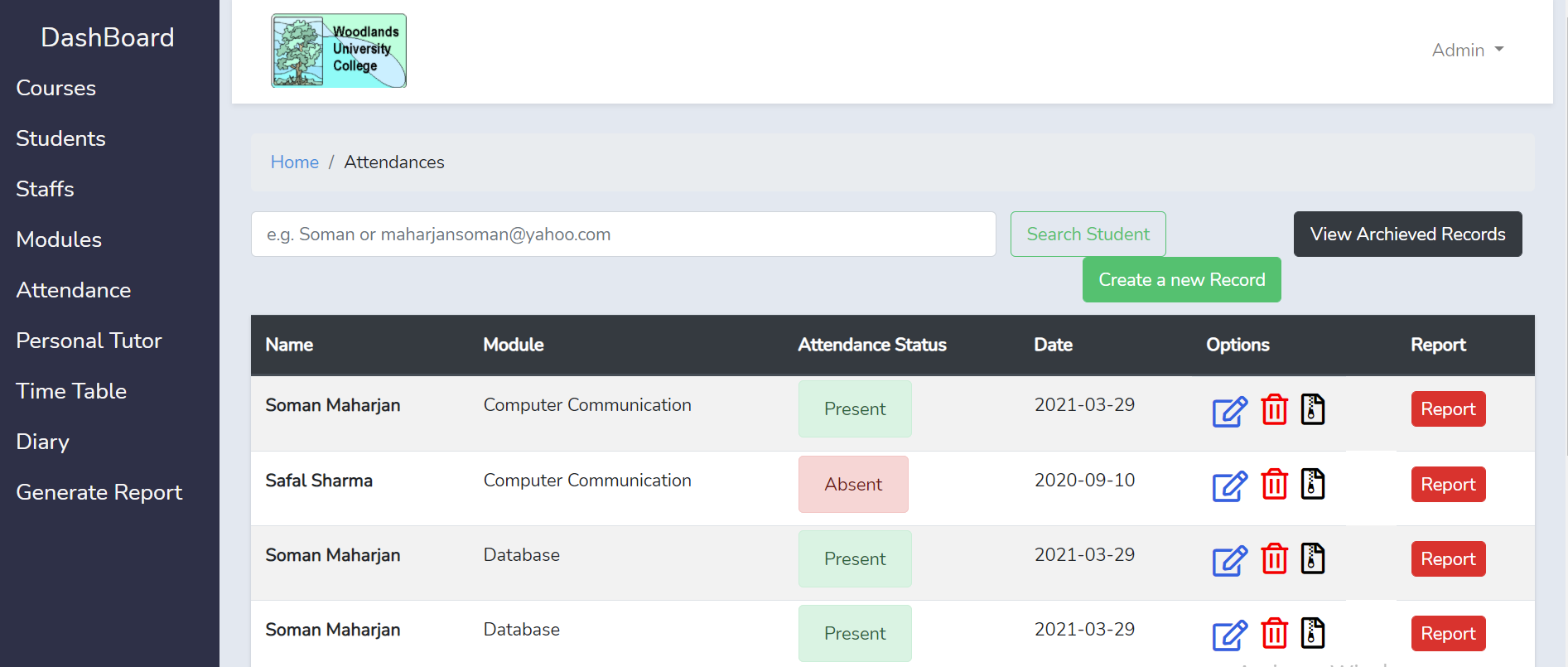


Figure 5.3.3: Attendance Page

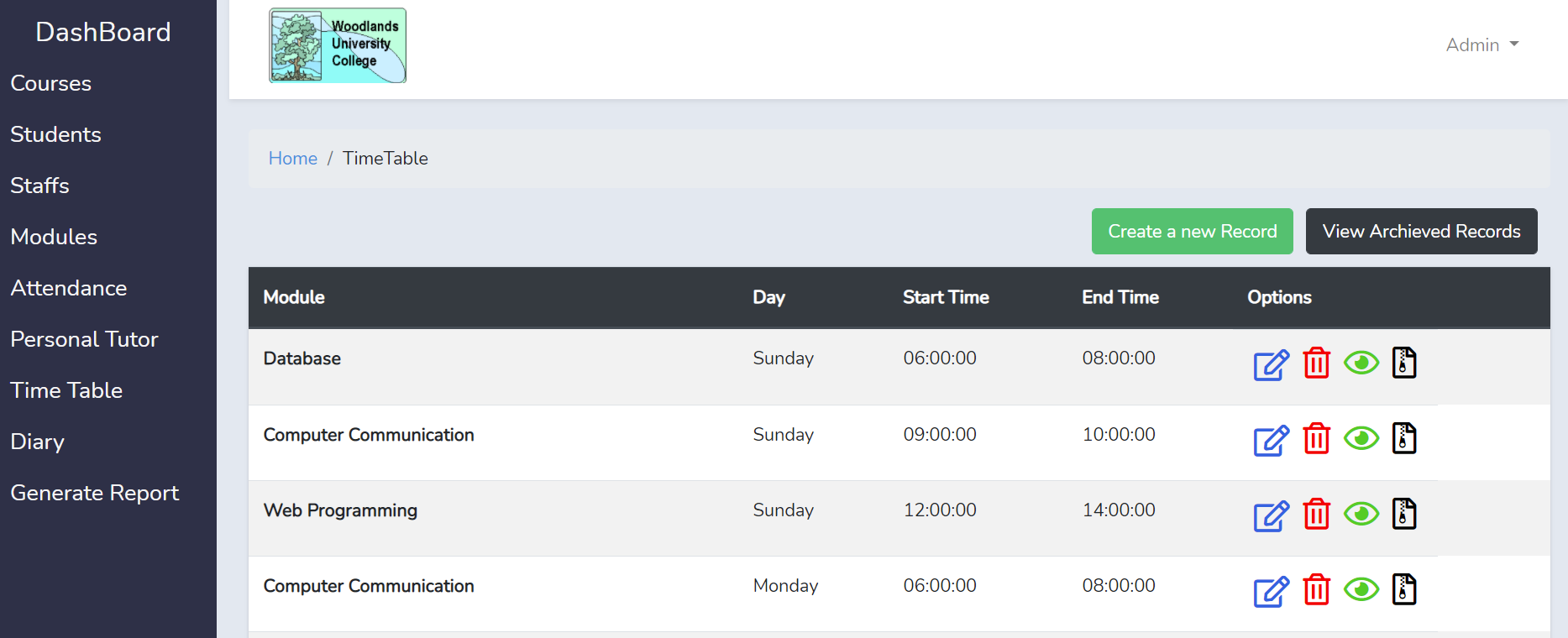


Figure 5.3.4: Timetable Page

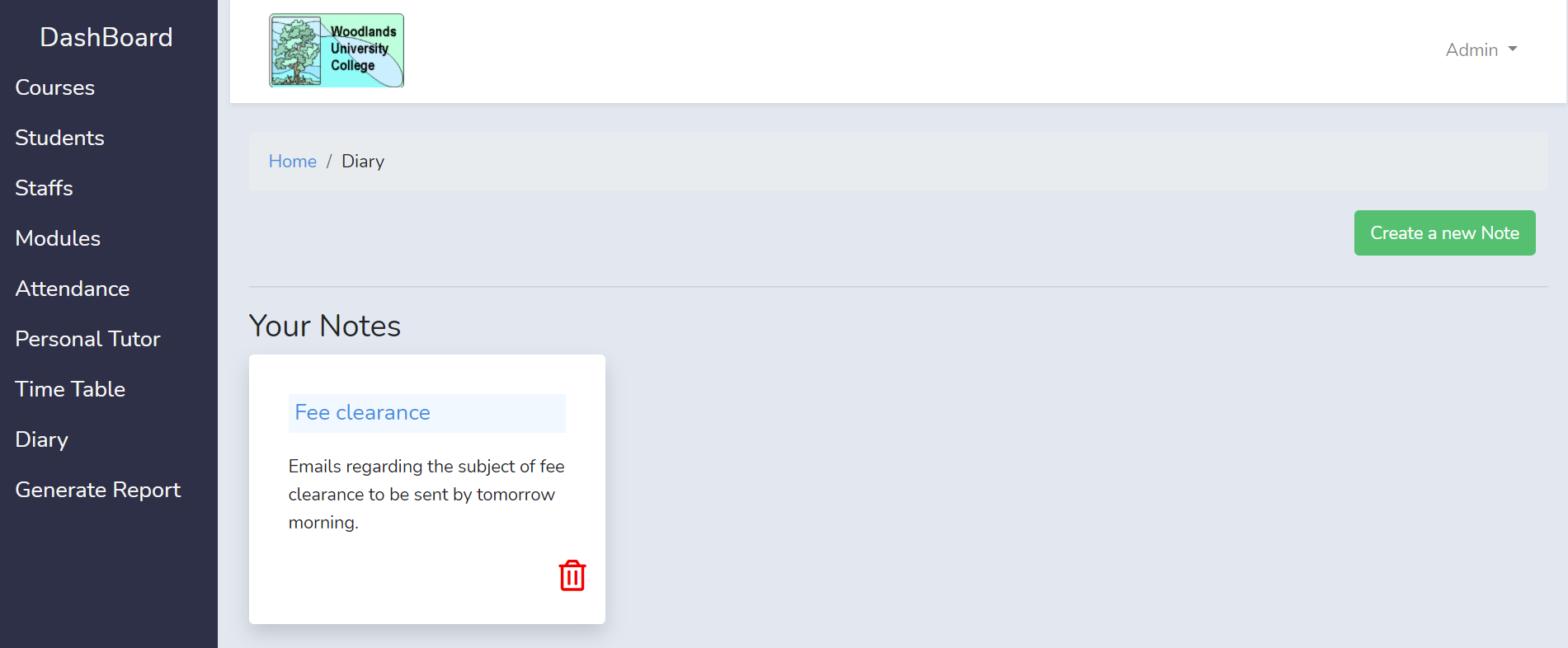


Figure 5.3.5: Diary Page

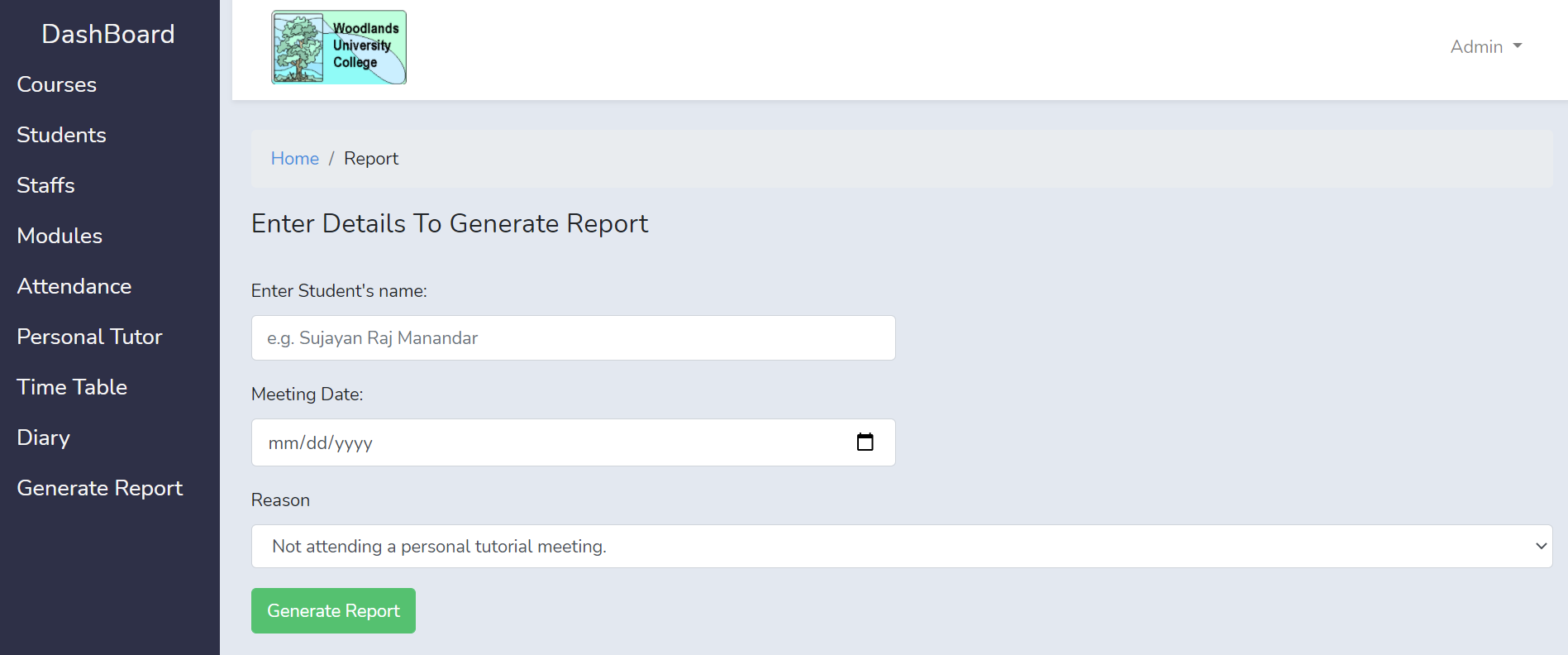


Figure 5.3.6: Generate Report Page

**Pages from Staff’s view:**

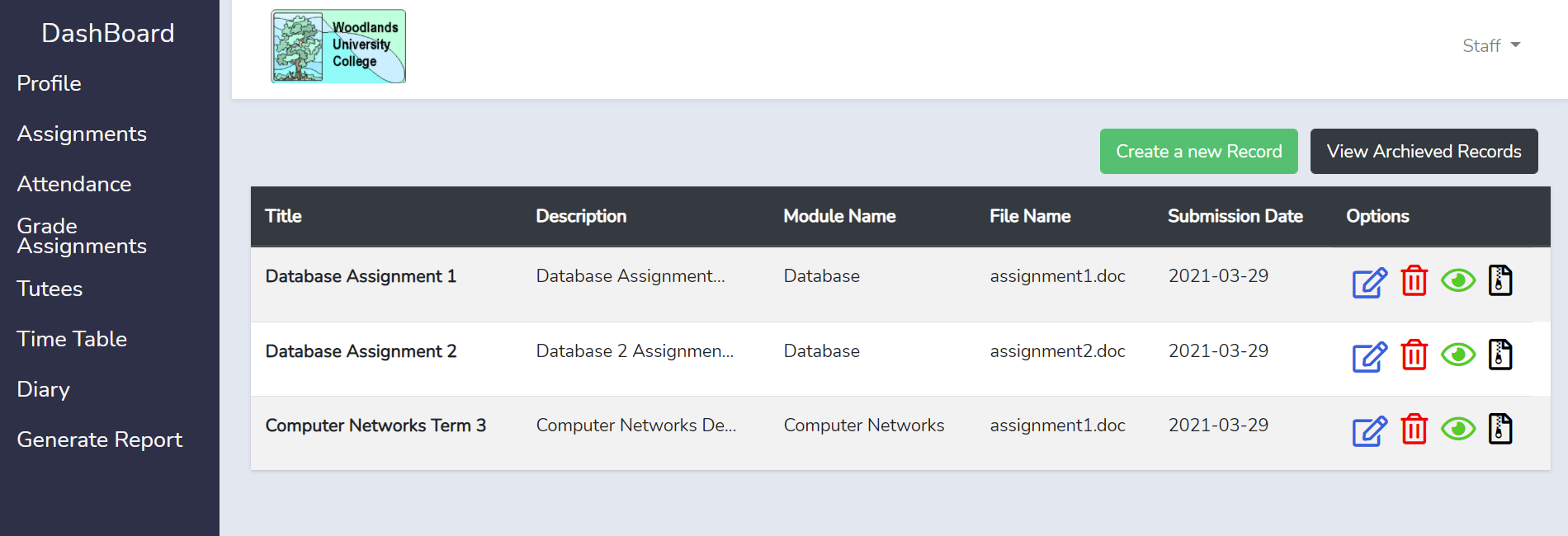


Figure 5.3.6: Assignments Page

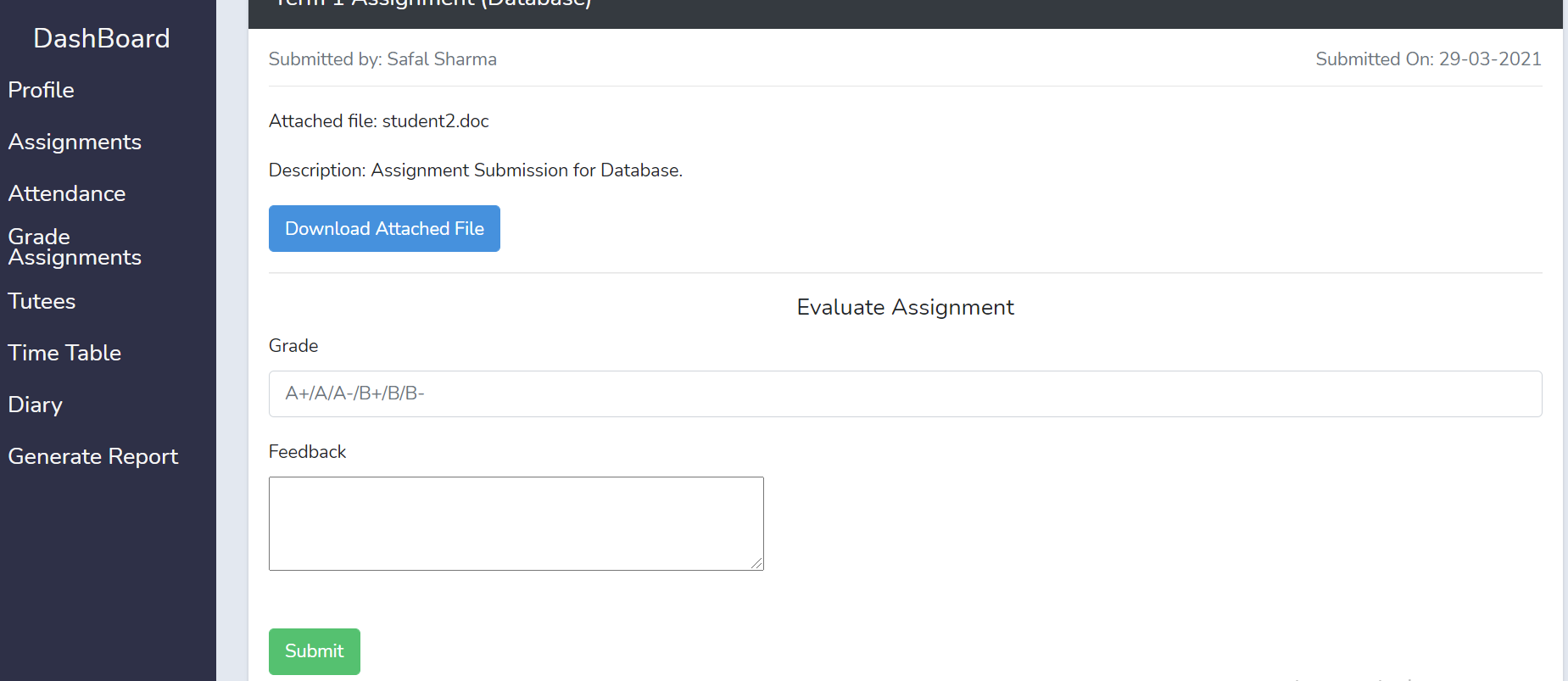


Figure 5.3.6: Grade Assignments Page

**Pages from Student’s view:**

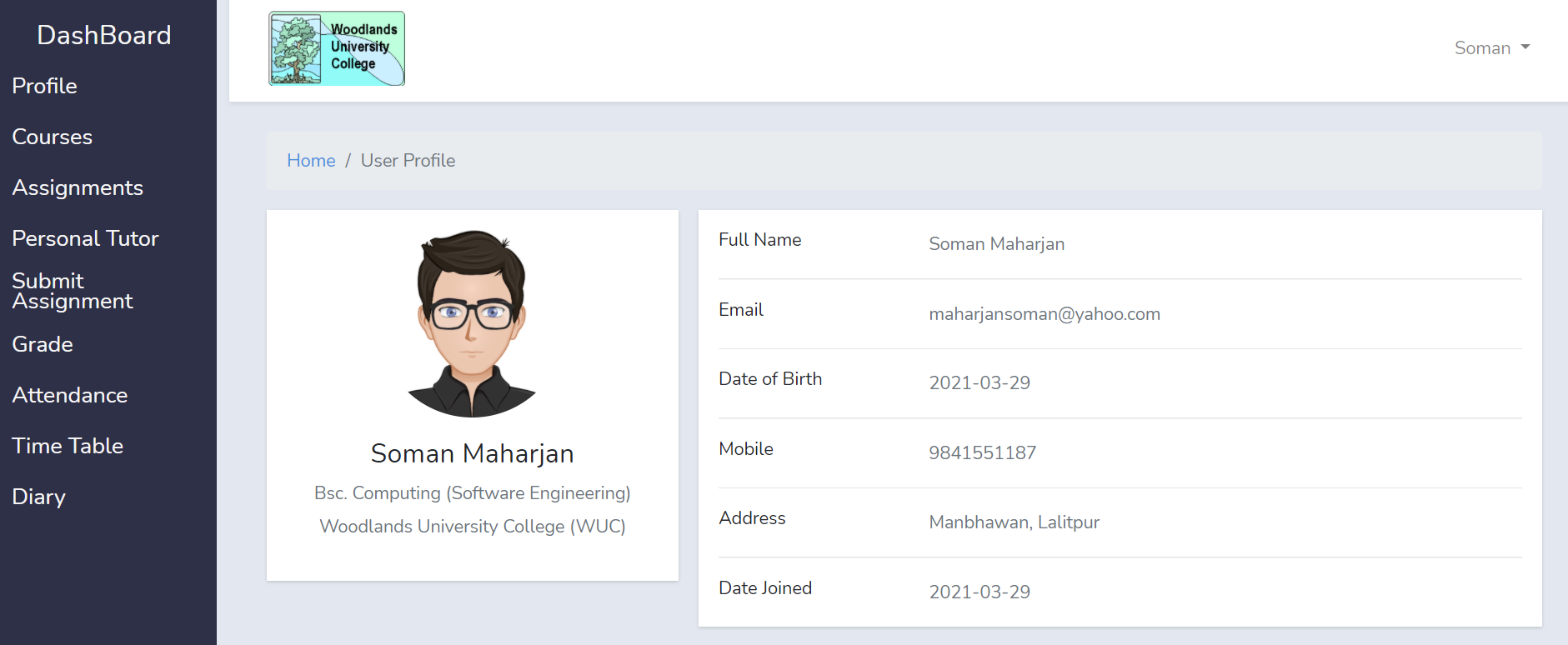


Figure 5.3.7: Student Profile Page

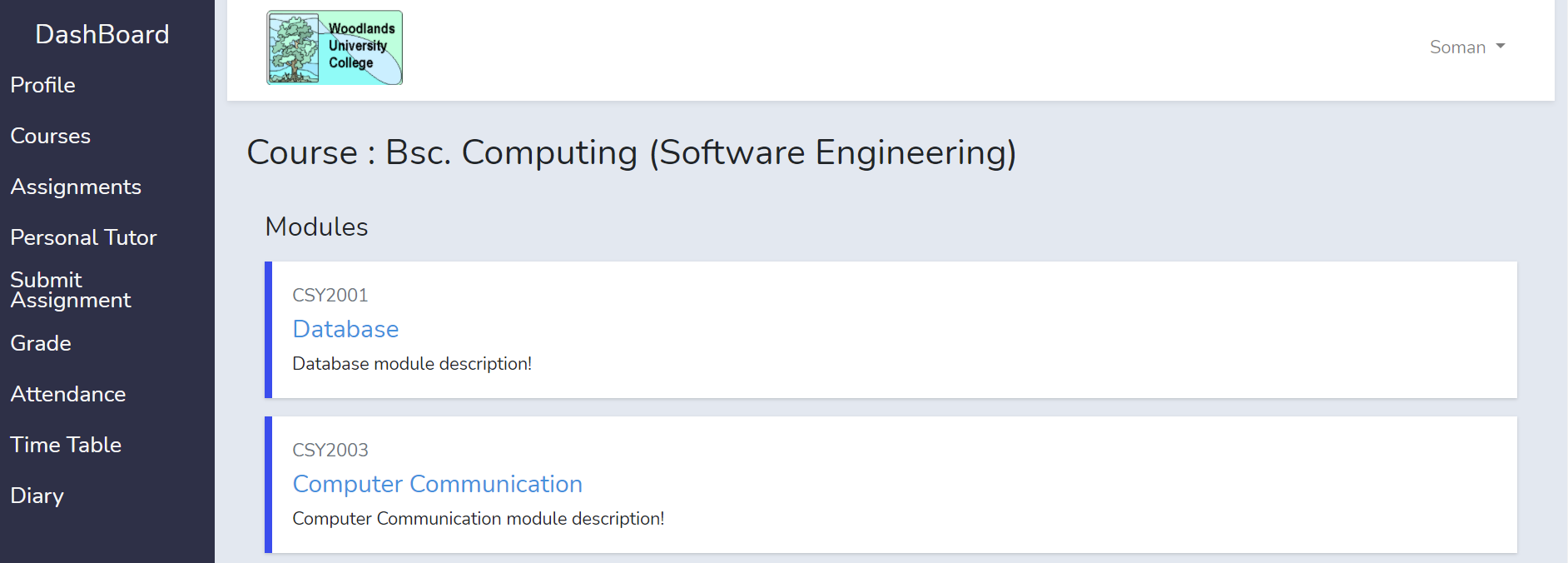


Figure 5.3.8: Courses Page

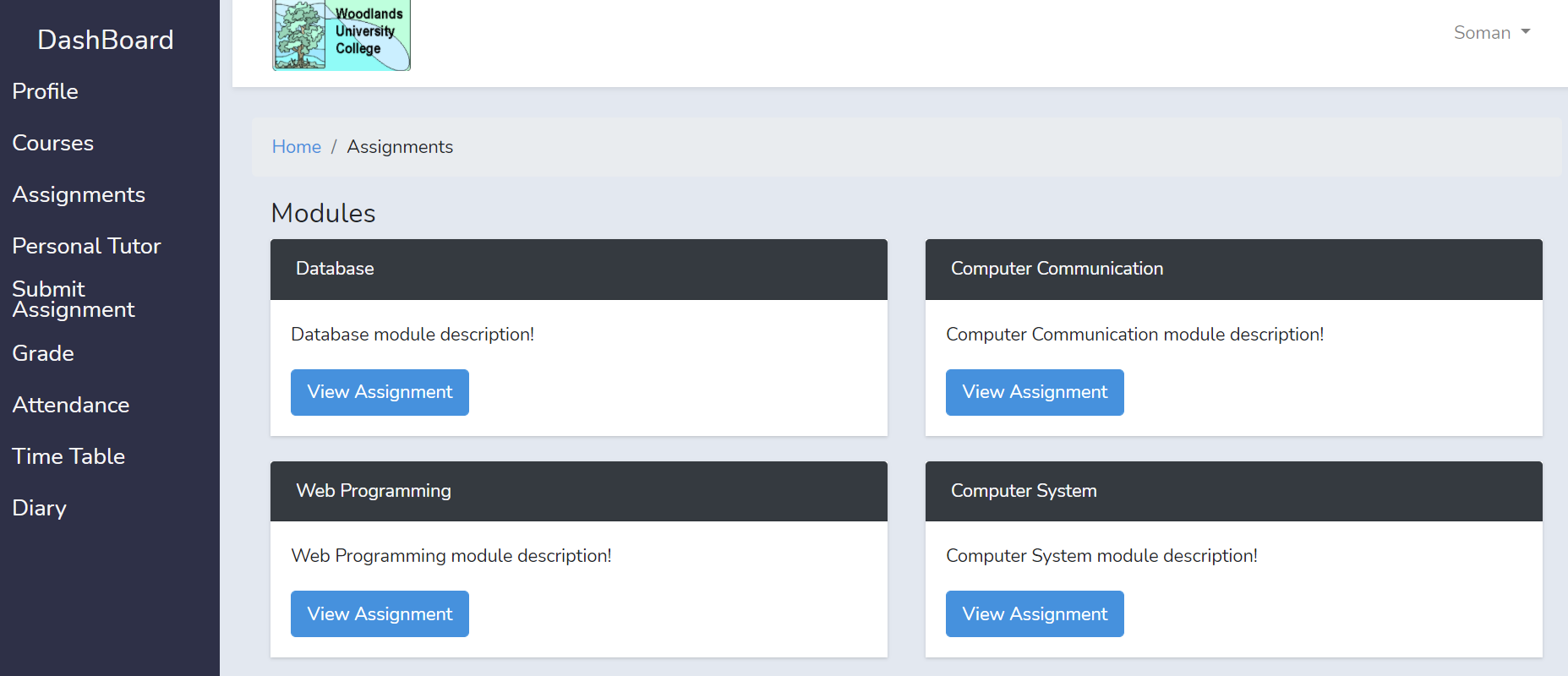


Figure 5.3.9: Assignments Page

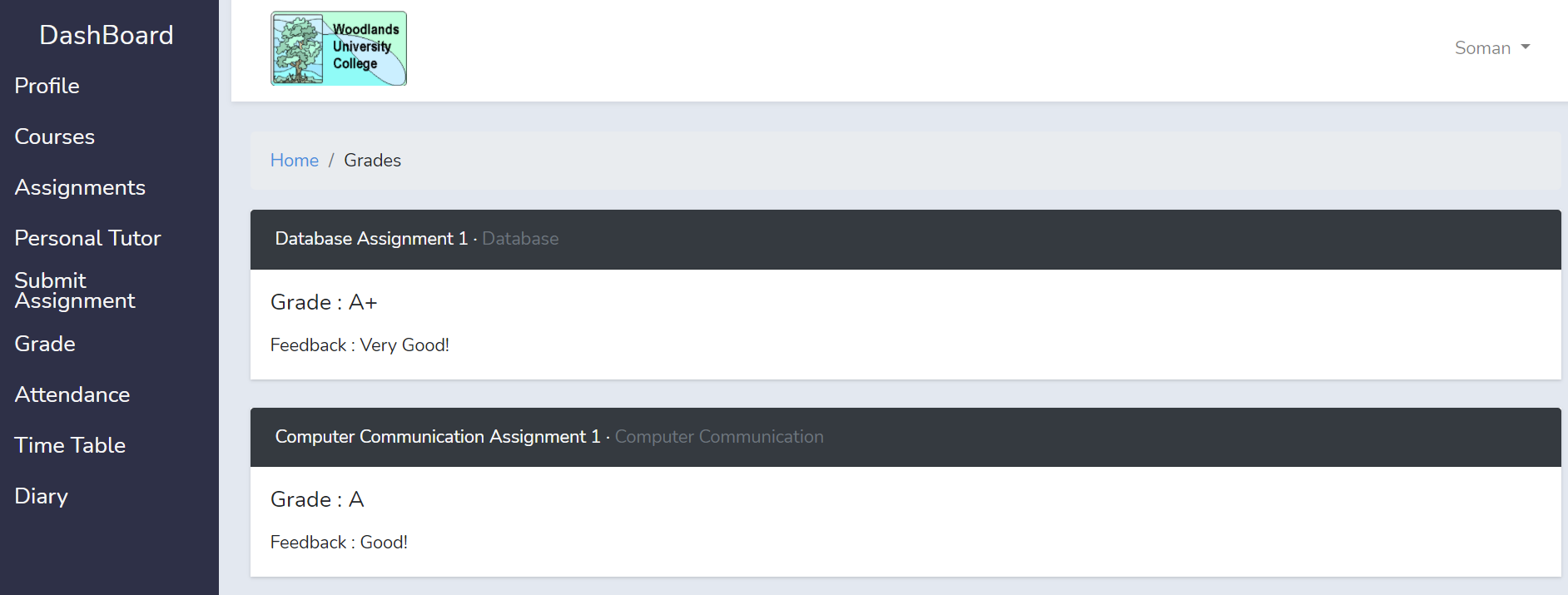


Figure 5.3.10: Grades Page

# 6 Test Strategy:

## Overview of Test Strategy:

Different test cases were created to evaluate various functionalities that could be used by admin, staff and student groups. Both valid and invalid data were supplied for the test cases to check how the system handles invalid data and processes valid data. Along with test cases, expected outputs were also formulated and later compared with actual outputs. The following table in 6.2 shows the results of testing, containing test data, actual and expected output as well as remarks which state whether the test was successful or not.

## Sample Test Results:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.N** | **Test Data** | **Process** | **Expected Output** | **Actual Output** | **Remarks** |
|  | **Admin** |  |  |  |  |
|  | Login with correct credentials. Example:  email: admin@admin.com  password: password | - Go to login page  - Fill credentials  - Press login | Successful login as admin | Admin homepage seen (successful login) | Test case successful |
|  | Login with incorrect credentials. Example:  email: staff@gmail.com  password: password | - Go to login page  - Fill credentials  - Press login | Login not successful | Stay in login page(error shown) | Test case successful |
|  | Press on courses on dashboard | - Dashboard displayed after successful login | All courses should be displayed | All courses displayed | Test case successful |
|  | Add a new course (with all fields filled correctly) | - Go to courses  - Press on create a new record button | New course should be displayed in the list | New course displayed in the list | Test case successful |
|  | Add a new course (incomplete form) | - Go to courses  - Press on create a new record button | Display error, record not submitted | New course is not added | Test case successful |
|  | Edit a course(change title) | - Go to courses  - Press on edit icon on a record | Display edited course name | Edited course name displayed | Test case successful |
|  | Delete a course | - Go to courses  - Press on delete icon on a record | Course should be removed from list of all courses | Course removed from list of all courses | Test case successful |
|  | Display a course | - Go to courses  - Press on display icon on a record | Information about course should be displayed | Information about course displayed | Test case successful |
|  | Archive a course | - Go to courses  - Press on archive icon on a record | Course should be removed from display | Course removed from display | Test case successful |
|  | View list of archived courses | - Go to courses  - Press on view archived results button | All archived courses should be displayed | All archived courses displayed | Test case successful |
|  | Unarchive a course | - Go to courses  - Press on view archived results button  - Press on unarchive icon on record | Course should be unarchived, displayed in list of all courses | Course unarchived, displayed in list of all courses | Test case successful |
|  | Press on students on dashboard | - Dashboard displayed after successful login | List of all students should be displayed | List of all students displayed | Test case successful |
|  | Add a new student (with all fields filled correctly) | - Go to students  - Press on create a new record button | New student should be displayed in the list | New student displayed in the list | Test case successful |
|  | Login with new student email and phone number as password | - Logout  - Redirected to login page  - Fill credentials  - Press login | Login should be successful | Student homepage seen (successful login) | Test case successful |
|  | Add a new student (incomplete form) | - Go to students  - Press on create a new record button | Display error, record not submitted | New student is not added | Test case successful |
|  | Edit a student record(change address) | - Go to students  - Press on edit icon on a record | Display edited student address | Edited student address displayed | Test case successful |
|  | Delete a student record | - Go to students  - Press on delete icon on a record | Student should be removed from list of all students | Student removed from list of all students | Test case successful |
|  | Display a student record | - Go to students  - Press on display icon on a record | Information about student should be displayed | Information about student displayed | Test case successful |
|  | Archive a student record | - Go to students  - Press on archive icon on a record | Student record should be removed from display | Student record removed from display | Test case successful |
|  | View list of archived student records | - Go to students  - Press on view archived results button | All archived student records should be displayed | All archived student records displayed | Test case successful |
|  | Unarchive a student record | - Go to students  - Press on view archived results button  - Press on unarchive icon on record | Student record should be unarchived, displayed in list of all students | Student record unarchived, displayed in list of all students | Test case successful |
|  | Press on staff on dashboard | - Dashboard displayed after successful login | List of all staff should be displayed | List of all staff displayed | Test case successful |
|  | Add a new staff (with all fields filled correctly) | - Go to staff  - Press on create a new record button | New staff should be displayed in the list | New staff displayed in the list | Test case successful |
|  | Login with new staff email and phone number as password | - Logout  - Redirected to login page  - Fill credentials  - Press login | Login should be successful | Staff homepage seen (successful login) | Test case successful |
|  | Add a new staff (incomplete form) | - Go to staff  - Press on create a new record button | Display error, record not submitted | New staff is not added | Test case successful |
|  | Edit a staff record(change job position) | - Go to staff  - Press on edit icon on a record | Display edited job position | Edited job position displayed | Test case successful |
|  | Delete a staff record | - Go to staff  - Press on delete icon on a record | Staff should be removed from list of all staff | Staff removed from list of all staff | Test case successful |
|  | Display a staff record | - Go to staff  - Press on display icon on a record | Information about staff should be displayed | Information about staff displayed | Test case successful |
|  | Archive a staff record | - Go to staff  - Press on archive icon on a record | Staff record should be archived and removed from display of all staff | Staff record archived, removed from display of all staff | Test case successful |
|  | View list of archived staff records | - Go to staff  - Press on view archived results button | All archived staff records should be displayed | All archived staff records displayed | Test case successful |
|  | Unarchive a staff record | - Go to staff  - Press on view archived results button  - Press on unarchive icon on record | Staff record should be unarchived, displayed in list of all staff | Staff record unarchived, displayed in list of all staff | Test case successful |
|  | Press on modules on dashboard | - Dashboard displayed after successful login | List of all modules should be displayed | List of all modules displayed | Test case successful |
|  | Add a new module (with all fields filled correctly) | - Go to modules  - Press on create a new record button | New module should be displayed in the list | New module displayed in the list | Test case successful |
|  | Add a new module (incomplete form) | - Go to modules  - Press on create a new record button | Display error, record not submitted | New module is not added | Test case successful |
|  | Edit a module record(change start date) | - Go to modules  - Press on edit icon on a record | Display edited start date | Edited start date displayed | Test case successful |
|  | Delete a module record | - Go to modules  - Press on delete icon on a record | Module should be removed from list of all modules | Module removed from list of all modules | Test case successful |
|  | Display a module record | - Go to modules  - Press on display icon on a record | Information about module should be displayed | Information about module displayed | Test case successful |
|  | Archive a module record | - Go to modules  - Press on archive icon on a record | Module record should be removed from display | Module record removed from display | Test case successful |
|  | View list of archived module records | - Go to modules  - Press on view archived results button | All archived module records should be displayed | All archived module records displayed | Test case successful |
|  | Unarchive a module record | - Go to modules  - Press on view archived results button  - Press on unarchive icon on record | Module record should be unarchived, displayed in list of all modules | Module record unarchived, displayed in list of all modules | Test case successful |
|  | Press on attendance on dashboard | - Dashboard displayed after successful login | List of all attendance records should be displayed | List of all attendance records displayed | Test case successful |
|  | Add a new attendance record (with all fields filled correctly) | - Go to attendance  - Press on create a new record button | New attendance record should be displayed in the list | New attendance record displayed in the list | Test case successful |
|  | Add a new attendance record (incomplete form) | - Go to attendance  - Press on create a new record button | Display error, record not submitted | New attendance record is not added | Test case successful |
|  | Edit an attendance record(change date) | - Go to attendance  - Press on edit icon on a record | Display edited date | Edited date displayed | Test case successful |
|  | Delete an attendance record | - Go to attendance  - Press on delete icon on a record | Attendance record should be removed from list of all staff | Attendance record removed from list of all staff | Test case successful |
|  | Search an attendance record | - Go to attendance  - Enter search term  - Press search | Information about attendance of particular student to be displayed | Information about attendance of particular student displayed | Test case successful |
|  | Archive an attendance record | - Go to attendance  - Press on archive icon on a record | Attendance record should be removed from display | Attendance record removed from display | Test case successful |
|  | View list of archived attendance records | - Go to attendance  - Press on view archived results button | All archived attendance records should be displayed | All archived attendance records displayed | Test case successful |
|  | Unarchive an attendance record | - Go to attendance  - Press on view archived results button  - Press on unarchive icon on record | Attendance record should be unarchived, displayed in list of all attendance records | Attendance record unarchived, displayed in list of all attendance records | Test case successful |
|  | Press on personal tutor on dashboard | - Dashboard displayed after successful login | List of all personal tutors should be displayed | List of all personal tutors displayed | Test case successful |
|  | Assign personal tutor to student | - Go to personal tutor  - Select student and tutor name from drop-down  - Press submit | Personal tutor should be assigned to student | Personal tutor assigned to student | Test case successful |
|  | Press on timetable on dashboard | - Dashboard displayed after successful login | List of all timetables should be displayed | List of all timetables displayed | Test case successful |
|  | Add a new timetable (with all fields filled correctly) | - Go to timetable  - Press on create a new record button | New timetable should be displayed in the list | New timetable displayed in the list | Test case successful |
|  | Add a new timetable (incomplete form) | - Go to timetable  - Press on create a new record button | Display error, record not submitted | New timetable is not added | Test case successful |
|  | Edit a timetable record(change start time) | - Go to timetable  - Press on edit icon on a record | Display edited start time | Edited start time displayed | Test case successful |
|  | Delete a timetable record | - Go to timetable  - Press on delete icon on a record | Timetable record should be removed from list of all timetables | Timetable record removed from list of all timetables | Test case successful |
|  | Display a timetable record | - Go to timetable  - Press on display icon on a record | Information about timetable should be displayed | Information about timetable displayed | Test case successful |
|  | Archive a timetable record | - Go to timetable  - Press on archive icon on a record | Timetable record should be archived, removed from display of all timetables | Timetable record archived, removed from display of all timetables | Test case successful |
|  | View list of archived timetable records | - Go to timetable  - Press on view archived results button | All archived timetable records should be displayed | All archived timetable records displayed | Test case successful |
|  | Unarchive a timetable record | - Go to timetable  - Press on view archived results button  - Press on unarchive icon on record | Timetable record should be unarchived, displayed in list of all timetable records | Timetable record unarchived, displayed in list of all timetable records | Test case successful |
|  | Press on diary on dashboard | - Dashboard displayed after successful login | List of all notes should be displayed | List of all notes displayed | Test case successful |
|  | Create a new note | - Go to diary  - Press on create a new note button  - Fill form  - Press submit | New note should be added | New note added | Test case successful |
|  | Delete a note | - Go to diary  - Press on delete icon on a note | Note should be deleted | Note deleted | Test case successful |
|  | Press on generate report on dashboard | - Dashboard displayed after successful login | Form to generate report should be displayed | Form to generate report displayed | Test case successful |
|  | Create a new report | - Go to generate report  - Select student and reason from drop-down, enter date  - Press generate | Created report should be displayed | Created report displayed | Test case successful |
|  | Print report | - Go to generate report  - Press print report | Print popup should be displayed | Print popup displayed | Test case successful |
|  | **Staff** |  |  |  |  |
|  | Login with correct credentials. Example:  email: staff@staff.com  password: password | - Go to login page  - Fill credentials  - Press login | Successful login as staff | Staff homepage seen (successful login) | Test case successful |
|  | Login with incorrect credentials. Example:  email: staff @gmail.com  password: password | - Go to login page  - Fill credentials  - Press login | Login not successful | Stay in login page(error shown) | Test case successful |
|  | Press on profile on dashboard | - Dashboard displayed after successful login | Profile with staff information should be displayed | Profile with staff information displayed | Test case successful |
|  | Press on assignments on dashboard | - Dashboard displayed after successful login | All assignments should be displayed | All assignments displayed | Test case successful |
|  | Add a new assignment (with all fields filled correctly) | - Go to assignments  - Press on create a new record button | New assignment should be displayed in the list | New assignment displayed in the list | Test case successful |
|  | Add a new assignment (incomplete form) | - Go to assignments  - Press on create a new record button | Display error, record not submitted | New assignment is not added | Test case successful |
|  | Edit an assignment (change title) | - Go to assignments  - Press on edit icon on a record | Display edited assignment title | Edited assignment title displayed | Test case successful |
|  | Delete an assignment | - Go to assignments  - Press on delete icon on a record | Assignment should be removed from list of all assignments | Assignment removed from list of all assignments | Test case successful |
|  | Display an assignment record | - Go to assignments  - Press on display icon on a record | Information about assignment should be displayed | Information about assignment displayed | Test case successful |
|  | Archive an assignment record | - Go to assignments  - Press on archive icon on a record | Assignment record should be removed from display | Assignment record removed from display | Test case successful |
|  | View list of archived assignment records | - Go to assignments  - Press on view archived results button | All archived assignment records should be displayed | All archived assignment records displayed | Test case successful |
|  | Unarchive an assignment record | - Go to assignments  - Press on view archived results button  - Press on unarchive icon on record | Assignment record should be unarchived, displayed in list of all assignment records | Assignment record unarchived, displayed in list of all assignment records | Test case successful |
|  | Press on tutees on dashboard | - Dashboard displayed after successful login | Information on the personal tutor’s tutees should be displayed | Information on the personal tutor’s tutees displayed | Test case successful |
|  | Press on grade assignments on dashboard | - Dashboard displayed after successful login | All submitted assignments should be displayed | All submitted assignments displayed | Test case successful |
|  | Grade an assignment (with all fields filled correctly) | - Go to grade assignments  - Fill form  - Press submit button | Assignment grade should be recorded | Assignment grade recorded | Test case successful |
|  | Grade an assignment (incomplete form) | - Go to grade assignments  - Fill form  - Press submit button | Display error, record not submitted | Assignment grade is not recorded | Test case successful |
|  | **Student** |  |  |  |  |
|  | Login with correct credentials. Example:  email: maharjansoman@yahoo.com  password: 9841551187 (email id and phone number of a student, added by admin) | - Go to login page  - Fill credentials  - Press login | Successful login as student | Student homepage seen (successful login) | Test case successful |
|  | Login with incorrect credentials. Example:  email: staff @gmail.com  password: password | - Go to login page  - Fill credentials  - Press login | Login not successful | Stay in login page(error shown) | Test case successful |
|  | Press on profile on dashboard | - Dashboard displayed after successful login | Profile with student information should be displayed | Profile with student information displayed | Test case successful |
|  | Press on courses on dashboard | - Dashboard displayed after successful login | Course name with all associated modules should be displayed | Course name with all associated modules displayed | Test case successful |
|  | Press on assignments on dashboard | - Dashboard displayed after successful login | All modules should be displayed with an option to view assignment | All modules displayed with an option to view assignment | Test case successful |
|  | Press on personal tutor on dashboard | - Dashboard displayed after successful login | Information on personal tutor should be displayed | Information on personal tutor displayed | Test case successful |
|  | Press on grade on dashboard | - Dashboard displayed after successful login | All grades should be displayed | All grades displayed | Test case successful |
|  | Press on submit assignment on dashboard | - Dashboard displayed after successful login | All forms to submit assignments should be displayed | All forms to submit assignments displayed | Test case successful |
|  | Submit an assignment (with all fields filled correctly) | - Go to submit assignment  - Fill form  - Press submit button | Assignment should be successfully submitted | Assignment successfully submitted | Test case successful |
|  | Submit an assignment (incomplete form) | - Go to submit assignment  - Fill form  - Press submit button | Display error, assignment not submitted | Assignment submission is not successful | Test case successful |
|  | Press on attendance on dashboard | - Dashboard displayed after successful login | Attendance record of student should be displayed | Attendance record of student displayed | Test case successful |
|  | Press on timetable on dashboard | - Dashboard displayed after successful login | Timetable of all days should be displayed | Timetable of all days displayed | Test case successful |
|  | View timetable on Sunday | - Go to timetable  - Select Sunday | Timetable of Sunday should be displayed | Timetable of Sunday displayed |  |
|  | Press on diary on dashboard | - Dashboard displayed after successful login | List of all notes should be displayed | List of all notes displayed | Test case successful |

# 7 System Evaluation:

## 7.1 Usability Evaluation Strategy:

System Evaluation is the set of actions taken to estimate the efficiency of a whole system in an outside traditional marketing environment. This strategy will be taken with the series of questionnaire asked to the people of Woodlands University College and asking them to use the system on their benefit and examine its usability and effectiveness.

The questions to be asked are as follows:

1. How much did you enjoy the experience with the system in general?

2. How were the system complication factors?

3. Can you perform the system functionalities smoothly?

4. Are all your intended functionalities work well?

5. Can you easily work with the system? Do you need guidance?

6. Did you encounter errors while working with the system?

7. Do you think all the people associated with the university can get used to using this system?

8. Can you see this system as all-user friendly?

9. Do you think that the system will be helpful for the university?

10. To what extent were you able to control the system?

11. How was the information provided by the system?

## 7.2 System Plot Trials:

The questionnaire experiment was done with some of the candidates of the university, which is knows as System Pilot Trials.

**Candidate 1:**

1. How much did you enjoy the experience with the system in general?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** |  |
| **Satisfactory** | ✓ |
| **Excellent** |  |

1. How were the system complication factors?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** | ✓ |
| **Satisfactory** |  |
| **Excellent** |  |

1. Can you perform the system functionalities smoothly?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** | ✓ |
| **Satisfactory** |  |
| **Excellent** |  |

1. Are all your intended functionalities work well?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. Can you easily work with the system? Do you need guidance?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. Did you encounter errors while working with the system?

|  |  |
| --- | --- |
| **Yes** |  |
| **No** | ✓ |

1. Do you think all the people associated with the university can get used to using this system?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. Can you see this system as all-user friendly?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. Do you think that the system will be helpful for the university?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. To what extent were you able to control the system?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** | ✓ |
| **Satisfactory** |  |
| **Excellent** |  |

1. How was the information provided by the system?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** |  |
| **Satisfactory** | ✓ |
| **Excellent** |  |

**Candidate 2:**

1. How much did you enjoy the experience with the system in general?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** |  |
| **Satisfactory** | ✓ |
| **Excellent** |  |

1. How were the system complication factors?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** | ✓ |
| **Satisfactory** |  |
| **Excellent** |  |

1. Can you perform the system functionalities smoothly?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** |  |
| **Satisfactory** |  |
| **Excellent** | ✓ |

1. Are all your intended functionalities work well?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. Can you easily work with the system? Do you need guidance?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. Did you encounter errors while working with the system?

|  |  |
| --- | --- |
| **Yes** |  |
| **No** | ✓ |

1. Do you think all the people associated with the university can get used to using this system?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. Can you see this system as all-user friendly?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. Do you think that the system will be helpful for the university?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. To what extent were you able to control the system?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** |  |
| **Satisfactory** | ✓ |
| **Excellent** |  |

1. How was the information provided by the system?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** |  |
| **Satisfactory** | ✓ |
| **Excellent** |  |

**Candidate 3:**

1. How much did you enjoy the experience with the system in general?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** |  |
| **Satisfactory** | ✓ |
| **Excellent** |  |

1. How were the system complication factors?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** | ✓ |
| **Satisfactory** |  |
| **Excellent** |  |

1. Can you perform the system functionalities smoothly?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** |  |
| **Satisfactory** |  |
| **Excellent** | ✓ |

1. Are all your intended functionalities work well?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. Can you easily work with the system? Do you need guidance?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. Did you encounter errors while working with the system?

|  |  |
| --- | --- |
| **Yes** |  |
| **No** | ✓ |

1. Do you think all the people associated with the university can get used to using this system?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. Can you see this system as all-user friendly?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. Do you think that the system will be helpful for the university?

|  |  |
| --- | --- |
| **Yes** | ✓ |
| **No** |  |

1. To what extent were you able to control the system?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** |  |
| **Satisfactory** | ✓ |
| **Excellent** |  |

1. How was the information provided by the system?

|  |  |
| --- | --- |
| **Bad** |  |
| **Could be Worse** |  |
| **Moderate** |  |
| **Satisfactory** | ✓ |
| **Excellent** |  |

## 7.3 System Trial Results:

From the system trial pilot above with the 3 candidates, the result is extracted which is as below:

|  |  |
| --- | --- |
| System Evaluation. | Evaluation Result in Average. |
| System Usability | Satisfactory |
| System Complexity | Moderate |
| System Functionalities | Excellent |
| Bugs and Errors | None |
| Assistance | Required |
| Overall Experience | Satisfactory |
| Learning | Moderate |
| System Adaptability | Alright |
| System Economy | Alright |
| System Control | Satisfactory |
| System Information Specialization | Satisfactory |

# 8 Project Conclusions:

Woodlands University College is a higher education institution offering several courses. Keeping in mind the importance of quality education, Dr. Simon, the course leader of the computing department believes that the current course management system is not efficient. Following the digitization trend, the project aims to virtualize the existing system as well as provide good services to all of the staff, students, and administrative body. With a simple GUI (Graphical user interface), the project focuses on improving the management of records, improving communication between users, and providing secure access to personal data.

The objective of the project revolves around enhancing the user experience. This includes regular updating of records, timely uploading of assignments and timetables also, proper organization of personal diaries. Comparing with the previous system, there will be no redundant data or permanent data loss. Functionalities of the site include:

* Store, display, edit, archive student and staff data
* Enroll students into respective courses and display information about the courses
* Assign students to modules based on the chosen course
* Create and display new/submitted assignments and display grades
* Record attendance of teachers and students, inform if attendance is not adequate
* Assign personal tutors to students
* View the timetable of classes
* Make diary entries
* Create, display, and print reports

After thorough analysis and requirement engineering, an SRS document was developed. After listing the positive aspects of the previous system, a base for the project was created. After the functional requirement, commercial constraints, performance constraints, technical and non-functional requirements were identified, the system interface development was initiated. By using wireframes for the front-end and BON charts for the overall system design, planning for the project was done. The application of the code was made using HTML, CSS, JavaScript, Laravel, and PHP. For testing, both black box and white box testing were conducted. Participants included regular users and programmers. This aided to identify syntax errors as well as how to improve the user’s experience. To maintain this experience, the software will be regularly updated, constantly solving bugs for the smooth running of the site.

# 9 Project Management:

## 9.1 Project Gantt Chart:

Gantt chart is a chart in which a series of horizontal lines shows the amount of work done in certain periods of time in relation to the amount planned for those periods. It assists in the planning and scheduling of project of all sizes.

There are 2 Gantt Charts below to show the amount of time elapsed from the start of one event to its finish in both documentation and implementation process.

GANTT CHART

**Documentation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Topics | Start Week | End Week | Total Weeks |  |
| Introduction | 1 2 1 | | | |
| Requirement Engineering | 2 6 4 | | | |
| System Interface Designs | 6 9 3 | | | |
| System Analysis and Design | 9 12 3 | | | |
| System Build and Notes | 12 13 1 | | | |
| Test Strategy | 13 16 3 | | | |
| System Evaluation | 16 17 1 | | | |
| Project Conclusion | 17 18 1 | | | |
| Project Management | 18 20 2 | | | |

Total Time Elapsed

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Weeks | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** |
| Introduction |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Requirement Engineering |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| System Interface Designs |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| System Analysis and Design |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| System Build and Notes |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Test Strategy |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| System Evaluation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Project Conclusion |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Project Management |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Fig: Gantt chart to show the time elapsed in documentation of the project.

|  |
| --- |
|  |

Start Week

|  |
| --- |
|  |

End Week

|  |
| --- |
|  |

Total Weeks

**Code and Implementation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Topics | Start Week | End Week | Total Weeks |  |
| User Interface | 10 12 2 | | | |
| Administrator | 13 15 2 | | | |
| Module Leader | 15 17 2 | | | |
| Student | 17 20 4 | | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Weeks | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** |
| User Interface |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Administrator |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Module Leader |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Student |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

|  |
| --- |
|  |

Start Week

|  |
| --- |
|  |

End Week

|  |
| --- |
|  |

Total Weeks

Fig: Gantt Chart to show the time elapsed in coding and implementation of the project.

## 9.2 Project Meeting Minutes:

**Meeting 1:**

Date: 13th December 2020

Time: 18:30

Venue: Google Meet

Duration: 1 hour

In Attendance: Bipashu, Rabika, Soman, Sujayan

Agenda: Work on table specification and ERD

Progress: First draft of table specification and ERD completed with agreement of all members

**Meeting 2:**

Date: 17th December 2020

Time: 18:30

Venue: Google Meet

Duration: 1 hour

In Attendance: Bipashu, Rabika, Soman, Sujayan

Agenda: Discuss interview questions, finalize wireframe

Progress: Finalized questions, assigned roles in interview, wireframe completed

**Meeting 3:**

Date: 22nd December 2020

Time: 17:30

Venue: Google Meet

Duration: 1.5 hour

In Attendance: Rabika, Sujayan

Agenda: Interview

Progress: interview completed

**Meeting 4:**

Date: 25th December 2020

Time: 18:30

Venue: Google Meet

Duration: 1 hour

In Attendance: Bipashu, Rabika, Soman, Sujayan

Agenda: Discuss interview findings and finalize mockups

Progress: Finalized findings and division of work on report and coding, mock ups completed

**Meeting 5:**

Date: 4th Jan 2021

Time: 18:15

Venue: Google Meet

Duration: 1 hour

In Attendance: Bipashu, Rabika, Soman, Sujayan

Agenda: Finalize part 1 and 2 of report

Progress: Finalize questions, assign roles in interview, front end begun

**Meeting 6:**

Date: 16th Jan 2021

Time: 17:00

Venue: Google Meet

Duration: 1.5 hour

In Attendance: Bipashu, Rabika, Soman, Sujayan

Agenda: Show client the progress made so far

Progress: Client satisfied with current progress, identification of features to be further improved on.

**Meeting 7:**

Date: 25th Jan 2021

Time: 18:00

Venue: Google Meet

Duration: 1 hour

In Attendance: Bipashu, Rabika, Soman, Sujayan

Agenda: Run check of partial site, Compare site with client’s needs

Progress: Site meets needs of client, discussion of further features to be added

**Meeting 8:**

Date: 2nd Feb 2021

Time: 18:00

Venue: Google Meet

Duration: 30 mins

In Attendance: Bipashu, Rabika, Soman, Sujayan

Agenda: Discussion of part 3 of report

Progress: Bon diagrams finalized

**Meeting 9:**

Date: 9th Feb 2021

Time: 18:00

Venue: Google Meet

Duration: 1 hour

In Attendance: Bipashu, Rabika, Soman, Sujayan

Agenda: Discussion of part 3 of report

Progress: part 3 finalized

**Meeting 10:**

Date: 16th Feb 2021

Time: 18:00

Venue: Google Meet

Duration: 1 hour

In Attendance: Bipashu, Rabika, Soman, Sujayan

Agenda: Project review

Progress: Checked relations, solved login bug, further debugging done

**Meeting 11:**

Date: 23rd Feb 2021

Time: 18:00

Venue: Google Meet

Duration: 1 hour

In Attendance: Bipashu, Rabika, Soman, Sujayan

Agenda: complete program first run

Progress: identified bugs, identified features left to be added

**Meeting 12:**

Date: 1st March 2021

Time: 18:30

Venue: Google Meet

Duration: 1 hour

In Attendance: Bipashu, Rabika, Soman, Sujayan

Agenda: Discussion of part 4 of report

Progress: part 4 finalized

**Meeting 13:**

Date: 10th March 2021

Time: 18:30

Venue: Google Meet

Duration: 30 hour

In Attendance: Bipashu, Rabika, Soman, Sujayan

Agenda: Discussion of class

Progress: Identified missed features

**Meeting 14:**

Date: 17th March 2021

Time: 18:00

Venue: Google Meet

Duration: 1 hour

In Attendance: Bipashu, Rabika, Soman, Sujayan

Agenda: Finalize program, discuss part 8 and 9 of report

Progress: Program made ready for testing, work division for part 8 and 9 done

**Meeting 15:**

Date: 24th March 2021

Time: 18:30

Venue: Google Meet

Duration: 45 mins

In Attendance: Bipashu, Rabika, Soman, Sujayan

Agenda: Finalize part 8 and 9 of report

Progress: Part 8 and 9 completed

## 9.3 Project Quality Plan / Strategy:

|  |  |  |  |
| --- | --- | --- | --- |
| SNo. | Functionality | Importance | Completion Status |
| Administrator | | | |
| 1 | Add/ edit/ delete records of staff | High | Completed |
| 2 | Archive/ unarchive records of staff | Low | Completed |
| 3 | Read all existing records of staff | High | Completed |
| 4 | Add/ edit/ delete records of student | High | Completed |
| 5 | Archive/ unarchive records of student | Low | Completed |
| 6 | Read all existing records of student | High | Completed |
| 7 | Assign student course/ module | High | Completed |
| 8 | Assign PAT to student | High | Completed |
| 9 | Add/ edit/ delete records of course and modules | High | Completed |
| 10 | Archive/ unarchive records of course and modules | Low | Completed |
| 11 | Read all existing records of course and modules | High | Completed |
| 12 | Assign course/ module leader | High | Completed |
| 13 | Add/ edit/ delete records of attendance | High | Completed |
| 14 | Archive/ unarchive records of attendance | Low | Completed |
| 15 | Read all existing records of attendance | High | Completed |
| 16 | Report poor attendance record of a student | Medium | Completed |
| 17 | Search for attendance records of student | Medium | Completed |
| 18 | Add/edit personal tutor records | Medium | Completed |
| 19 | Display all available personal tutors | High | Completed |
| 20 | Add/edit/ delete note | Medium | Completed |
| 21 | Display all personal notes | Medium | Completed |
| 22 | Create report record | High | Completed |
| 23 | Display report record | High | Completed |
| 24 | Print report | High | Completed |
| 25 | Add/ edit/ delete assignment record | High | Completed |
| 26 | Display all assignment record | High | Completed |
| 27 | Archive/ unarchive assignment records | Low | Completed |
| 28 | Add/edit/delete timetable record | High | Completed |
| 29 | Display timetable records | High | Completed |
| 30 | Archive/unarchive timetable records | Low | Completed |
| Staff | | | |
| 31 | Post assignment of specific module | High | Completed |
| 32 | Grade assignments | High | Completed |
| 33 | Display graded assignments | High | Completed |
| 34 | View personal profile | Low | Completed |
| 35 | Read all existing records of attendance | High | Completed |
| 36 | Report poor attendance record of a student | Medium | Completed |
| 37 | Search for attendance records of student | Medium | Completed |
| Student | | | |
| 38 | View posted assignments | High | Completed |
| 39 | Submit assignments | High | Completed |
| 40 | Display grades of submitted assignments | High | Completed |
| 41 | View personal profile | Low | Completed |
| 42 | View timetable | High | Completed |
| 43 | Create note | Medium | Completed |
| 44 | Delete note | Medium | Completed |

## 9.4 Project Work Log:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task** | **Contribution (in %) of all members** | | | |
| **1) Introduction** | Bipashu Thakuri | Rabika Pradhananga | Soman Maharjan | Sujayan Raj Manandhar |
| 1.1 Project Background |  |  |  | 100% |
| 1.2 Project Aims and Objectives |  | 50% |  | 50% |
| 1.3 Project Development Methodology | 100% |  |  |  |
| **2) Requirements Engineering** | Bipashu Thakuri | Rabika Pradhananga | Soman Maharjan | Sujayan Raj Manandhar |
| 2.1 Elicitation Activities    2.1.1 Interview plans | 25% | 25% | 25% | 25% |
| 2.1.2 Interview findings |  | 50% |  | 50% |
| 2.1.3 Other problem domain research | 100% |  |  |  |
| 2.1.3.1 Comparable Software System Review | 50% |  | 50% |  |
| 2.1.3.2 Development Relevant Legislation | 100% |  |  |  |
| 2.1.3.3 Any other relevant problem domain investigation data |  |  | 50% | 50% |
| 2.2 Requirements Specification |  |  | 100% |  |
| 2.2.1 Problem Domain Description |  |  |  | 100% |
| 2.2.1.1 Existing Business Operation |  |  | 100% |  |
| 2.2.1.2 Summary of existing business limitations requiring resolution |  |  |  | 100% |
| 2.2.2 Functional Requirements | 100% |  |  |  |
| 2.2.3 Performance Requirements | 100% |  |  |  |
| 2.2.3.1  Speed |  |  |  | 100% |
| 2.2.3.2  Capacity | 100% |  |  |  |
| 2.2.3.3  Reliability |  | 100% |  |  |
| 2.2.3.4  Usability |  |  | 100% |  |
| 2.2.4 Design Constraints |  | 100% |  |  |
| 2.2.5 Commercial Constraints |  |  |  | 100% |
| 2.2.6 Acceptance Tests |  |  | 100% |  |
| **3) System Analysis & Design** | Bipashu Thakuri | Rabika Pradhananga | Soman Maharjan | Sujayan Raj Manandhar |
| 3.1 Preliminary Design Stages  3.1.1 Textual Analysis |  | 50% | 50% |  |
| 3.1.2 Significant Event Analysis |  |  | 100% |  |
| 3.1.3 Commands Queries and Constraints |  | 100% |  |  |
| 3.2 Detailed Static System Designs  3.2.1 First Draft BON System Architecture Diagram |  | 50% |  | 50% |
| 3.2.2 BON System Chart |  |  |  | 100% |
| 3.2.3 BON Cluster Charts |  | 100% |  |  |
| 3.2.4 BON Class Charts |  | 100% |  |  |
| 3.3 Detailed Dynamic System Designs  3.3.1 Events Charts | 100% |  |  |  |
| 3.3.2 Object Creation Charts | 50% |  | 50% |  |
| 3.3.3 System Scenario Charts |  |  | 100% |  |
| 3.3.4 Dynamic Diagrams | 100% |  |  |  |
| 3.4 System Database Design |  | 100% |  |  |
| 3.4.1 E-R Model |  | 100% |  |  |
| 3.4.2 Attribute Listings |  |  | 100% |  |
| **4) System Interface Designs** | Bipashu Thakuri | Rabika Pradhananga | Soman Maharjan | Sujayan Raj Manandhar |
| 4.1 Draft Interface Designs  4.1.1 Wireframes |  | 100% |  |  |
| 4.1.2 System Navigation Diagram |  |  |  | 100% |
| 4.1.3 System Screen mock-ups |  |  |  | 100% |
| 4.1.4 System Activity Event Diagrams | 50% | 50% |  |  |
| **5) System Build and Technical Notes** | Bipashu Thakuri | Rabika Pradhananga | Soman Maharjan | Sujayan Raj Manandhar |
| 5.1 Technically Difficult Code Sections |  |  | 100% |  |
| 5.2 On-going Testing Methodology | 50% | 50% |  |  |
| 5.3 Final System Interface Displays |  |  | 50% | 50% |
| **6) Test Strategy** | Bipashu Thakuri | Rabika Pradhananga | Soman Maharjan | Sujayan Raj Manandhar |
| 6.1 Overview of Test Strategy | 25% | 25% | 25% | 25% |
| 6.2 Sample Test Results | 25% | 25% | 25% | 25% |
| **7) System Evaluation** | Bipashu Thakuri | Rabika Pradhananga | Soman Maharjan | Sujayan Raj Manandhar |
| 7.1 Usability Evaluation Strategy | 100% |  |  |  |
| 7.2 System Pilot Trials |  | 50% | 50% |  |
| 7.3 System Trial Results | 25% | 25% | 25% | 25% |
| **8) Project Conclusions** | Bipashu Thakuri | Rabika Pradhananga | Soman Maharjan | Sujayan Raj Manandhar |
| Conclusion |  |  |  | 100% |
| **9) Project Management** | Bipashu Thakuri | Rabika Pradhananga | Soman Maharjan | Sujayan Raj Manandhar |
| 9.1 Project Gantt Chat/WBDS/Activity List | 50% |  | 50% |  |
| 9.2 Project Meeting Minutes | 25% | 25% | 25% | 25% |
| 9.3 Project Quality Plan/Strategy | 50% |  | 50% |  |
| 9.4 Project Work Log |  | 50% |  | 50% |
| **10) Code and Testing** | Bipashu Thakuri | Rabika Pradhananga | Soman Maharjan | Sujayan Raj Manandhar |
| Front-end | 50% |  |  | 50% |
| Back-end |  | 50% | 50% |  |
| Testing | 25% | 25% | 25% | 25% |