

ABSTRACT

This proposal outlines the development of a web-based employee appraisal system tailored for academic institutions, with the goal of improving the efficiency, accuracy, and consistency of academic staff performance evaluations. Traditional appraisal methods are often manual, inconsistent, and prone to delays in tracking staff accomplishments. To address these limitations, the proposed system aims to automate the appraisal workflow, reduce administrative workload, and ensure standardized evaluation based on designated key performance indicators (KPIs).

The system integrates core functionalities such as secure login authentication, dynamic performance entry submission by staff, administrative privilege assignment, profile management, and automated report generation. Superiors will be able to assess academic staff based on multiple performance areas, such as teaching, research, and publications, with support for custom weightages based on designation or faculty. Staff can view their appraisal progress and update accomplishments periodically.

The project uses the Rapid Application Development (RAD) methodology, which emphasizes iterative prototyping and continuous user involvement. This approach allows for rapid feedback integration and flexibility in feature adjustments. The system is developed using the MERN stack (MongoDB, Express.js, React.js, and Node.js), enabling both robust backend logic and a responsive user interface.

Area of Study: Web Application Development, Performance Evaluation Systems

Keywords: Employee Appraisal System, Academic Staff Evaluation, Performance Management, MERN Stack Development, Rapid Application Development (RAD)

TABLE OF CONTENTS

TITLE PAGE	i
ACKNOWLEDGMENT	ii
COPYRIGHT STATEMENT	iii
ABSTRACT	iv
TABLE OF CONTENTS	v
LIST OF FIGURES	viii
LIST OF TABLES	xi
LIST OF ABBREVIATIONS	xiii
CHAPTER 1 PROJECT BACKGROUND	1
1.1 Background Information	1
1.2 Problem Statement and Motivation	2
1.3 Project Objectives	3
1.4 Project Scope	4
1.5 Impact, significant and contribution	6
1.6 Summary	7
CHAPTER 2 LITERATURE REVIEW	8
2.1 Overview	8
2.2 Similar Projects	8
2.2.1 Online Fuzzy Based Performance Appraisal System [4]	8
2.2.2 Integrated Performance Appraisal System with Management by Objective Method [5]	10
2.3 Existing Websites	13
2.3.1 Employee Performance Evaluation System in PHP/MySQLi [6]	13
2.3.2 PeopleHR's performance appraisal system [7]	16
2.3.3 Trakstar – Employee Performance Appraisal System [8]	19
2.4 Summary	23

CHAPTER 3 Proposed Method/Approach	25
3.1 Overview	25
3.2 Design Specification	25
3.2.1 Methodologies and General Work Procedures	25
3.2.1.1 Requirements Planning Phase	25
3.2.1.2 User Design Phase	26
3.2.1.3 Construction Phase	26
3.2.1.4 Cutover Phase	27
3.2.2 Tools to Use	27
3.3 System Design/Overview	30
3.3.1 Use-Case Diagram	30
3.3.2 Use-Cases Description	31
3.3.3 Program Development	43
3.3.3.1 Server-side Development	43
3.3.3.2 Login Function Development	46
3.3.3.3 Faculty and Department Management Function Development	48
3.3.3.4 Staff Management Function Development	50
3.3.3.5 Viewable Staff Function Development	52
3.4 Implementation Issues and Challenges	53
3.5 Timeline	53
3.5.1 Overview	53
3.5.2 Gantt Chart	55
3.6 Summary	56
 CHAPTER 4 PRELIMINARY WORK	 57
4.1 Overview	57
4.2 Login Page	57
4.3 User Profile	58
4.3.1 Staff and Superior Profile	58
4.3.2 Administrator Profile	59
4.4 Manage Faculty and Department	60
4.5 Manage Staff	61

4.6 Manage Performance Area Weightage	63
4.7 View Performance Report	64
4.7.1 Staff Panel	64
4.7.2 Superior Panel	65
4.8 Add and View Performance Area Entry	65
4.9 Viewable Staff Page (Superior)	66
4.10 Summary	68
 CHAPTER 5 CONCLUSION	 69
 REFERENCES	 70
 APPENDIX A	
A.1 Poster	A-1

LIST OF FIGURES

Figure Number	Title	Page
Figure 2.2.1.1	Architecture of the proposed Fuzzy Based Performance Appraisal System [4]	9
Figure 2.2.1.2	Decision Variables Considered for Appraising Academic Staff Performance [4]	9
Figure 2.2.2.1	Appraisal System Process using MBO technique [5]	11
Figure 2.3.1.1	Log in page [6]	14
Figure 2.3.1.2	Admin's main page [6]	14
Figure 2.3.1.3	Admin's add evaluator page [6]	14
Figure 2.3.1.4	Evaluator's (superior) evaluation page [6]	15
Figure 2.3.1.5	Employee's task list page [6]	15
Figure 2.3.2.1	Manager's main page [7]	17
Figure 2.3.2.2	Manager's tasks page [7]	17
Figure 2.3.2.3	Manager's evaluation page [7]	18
Figure 2.3.2.4	Employee's main page [7]	18
Figure 2.3.2.5	Employee's planner page [7]	18
Figure 2.3.2.6	Employee's evaluation page [7]	19
Figure 2.3.3.1	Manager's tasks list page [8]	20
Figure 2.3.3.2	Manager's evaluation page for selected employees [8]	21
Figure 2.3.3.3	Administrator's appraisal page [8]	21
Figure 2.3.3.4	Administrator's appraisal page (Search for certain year appraisal list) [8]	22
Figure 2.3.3.5	Administrator's performance history page for selected department [8]	22
Figure 3.2.1.1	RAD Methodology [9]	25
Figure 3.3.1.1	Use-case Diagram for Comprehensive Employee Appraisal System	30
Figure 3.3.3.1.1	package.json (Frontend)	44
Figure 3.3.3.1.2	package.json (Backend)	45

Figure 3.3.3.1.3	routes Folder and authRoute.js	45
Figure 3.3.3.1.4	mongodb.js	45
Figure 3.3.3.1.5	server.js	46
Figure 3.3.3.2.1	Login.jsx (Handling login/signup submission)	47
Figure 3.3.3.2.2	authController.js (Login function)	47
Figure 3.3.3.2.3	authMiddleware.js	48
Figure 3.3.3.3.1	Faculties.jsx (with function to handle operation for department)	49
Figure 3.3.3.3.2	facultyController.js	49
Figure 3.3.3.3.3	departmentController.js	50
Figure 3.3.3.4.1	academic.jsx (Search staff)	51
Figure 3.3.3.4.2	staffController.js (Delete and Search controller function)	51
Figure 3.3.3.5.1	Staff.jsx (fetchViewableStaff function)	52
Figure 3.5.2.1	Gantt Chart of the Project Timeline	55
Figure 4.2.1	Login Page	58
Figure 4.3.1.1	Profile Management Page	59
Figure 4.3.1.2	Change Password in Profile Management Page	59
Figure 4.3.2.1	Administrator Profile Page	60
Figure 4.4.1	Faculties Page	61
Figure 4.4.2	Assign Privilege Form	61
Figure 4.5.1	Academic Staff Page	62
Figure 4.5.2	View Profile of Academic Staff	62
Figure 4.5.3	View Academic Staff Performance Report	63
Figure 4.6.1	Performance Area Weightage Adjustment Page	63
Figure 4.7.1.1	Staff Performance Report (Summary)	64
Figure 4.7.1.2	Staff Performance Report (Area Progress)	64
Figure 4.7.1.3	Staff Performance Report (Area Progress in Bar Chart)	65
Figure 4.8.1	Performance Entry Main Page	66
Figure 4.8.2	Form of Performance Entry (Publication)	66
Figure 4.9.1	View All Staff Page (Superior)	67

LIST OF TABLES

Table Number	Title	Page
Table 2.4.1	Comparison of strengths and weaknesses of existing websites	24
Table 3.2.2.1	The hardware component and requirements for website development	27
Table 3.2.2.2	The software component and requirements for website development	28
Table 3.3.2.1	Login Use Case Description	31
Table 3.3.2.2	Reset Password Use Case Description	32
Table 3.3.2.3	Change Password Use Case Description	32
Table 3.3.2.4	View Profile Use Case Description	33
Table 3.3.2.5	Edit Profile Use Case Description	34
Table 3.3.2.6	Add Performance Use Case Description	34
Table 3.3.2.7	View Performance Report Use Case Description (Staff / Superior)	35
Table 3.3.2.8	View Performance Report Use Case Description (Administrator)	35
Table 3.3.2.9	View Subordinate Use Case Description	36
Table 3.3.2.10	View Subordinate Profile Use Case Description	36
Table 3.3.2.11	View Subordinate Performance Report Use Case Description	37
Table 3.3.2.12	View Subordinate Entries Summary Use Case Description	37
Table 3.3.2.13	Add Faculty Use Case Description	38
Table 3.3.2.14	Edit Faculty Use Case Description	38
Table 3.3.2.15	Delete Faculty Use Case Description	39
Table 3.3.2.16	Add Department Use Case Description	39
Table 3.3.2.17	Edit Department Use Case Description	40
Table 3.3.2.18	Delete Department Use Case Description	40
Table 3.3.2.19	Add Staff Use Case Description	41
Table 3.3.2.20	Delete Staff Use Case Description	41

Table 3.3.2.21	Assign Privilege Use Case Description	42
Table 3.3.2.22	Disable Privilege Use Case Description	42
Table 3.3.2.23	Edit Performance Area Weightage Use Case Description	43

LIST OF ABBREVIATIONS

<i>PA</i>	Performance Appraisal
<i>KPI</i>	Key Performance Indicators
<i>HR</i>	Human Resource
<i>HTML</i>	Hypertext Markup Language
<i>CSS</i>	Cascading Style Sheets
<i>HTTP</i>	Hypertext Transfer Protocol
<i>AJAX</i>	Asynchronous JavaScript & XML
<i>JSON</i>	JavaScript Object Notation
<i>MATLAB</i>	Matrix Laboratory
<i>KB</i>	Knowledge Base
<i>MBO</i>	Management by Objective
<i>BARS</i>	Behaviourally Anchored Rating Scales
<i>RAD</i>	Rapid Application Development
<i>SDLC</i>	Software Development Life Cycle
<i>JWT</i>	JSON Web Token

CHAPTER 1 INTRODUCTION

1.1 Background Information

The quality of employees determines the success of an organization. The employees represent a crucial factor in any organizational context, functioning as the organizational entity's core. It is difficult for organizations to achieve their goals and objectives without them.

Performance appraisal (PA) is defined as a periodic review of the performance and contribution of an employee to the organization [1]. It is a formal and structured process utilized by organizations to assess and evaluate the performance of employees and provide actionable feedback on strengths and weaknesses, relative value, and potential for future development of employees [2], [3]. In this process, the employee's performance is judged by those with a higher level based on pre-defined standards or criteria. It serves as an important tool in evaluating, guiding, and enhancing the performance of employees to align with organizational goals. Besides, PA systems provide constructive criticism, set targets for the future, find out what people need in terms of growth as well as acknowledge accomplishments hence enhancing employee productivity and growth while contributing to organizational competitiveness.

As a consequence, a website is proposed to improve the overall evaluation process and reduce the workload of the superiors who take responsibility for evaluating their subordinate's performance and contribution. With this website, the superior can access and review the accomplishments of each subordinate. They can also view the appraisal rating level of the subordinates that is ranked by the system based on the pre-defined standards. In addition, the website also came up with a function that enables superiors to view the subordinate's timetables, allowing them to directly schedule time to have a meeting. This function ensures that the meeting is planned efficiently by avoiding any overlap in hours. Furthermore, this website provides reporting tools that assist the superiors in understanding the subordinate's progress over time to supervise whether they met Key Performance Indicators (KPI).

1.2 Problem Statement and Motivation

The appraisal system is an important tool to evaluate employees' performance in any organization. However, the traditional method used to rate the performance and contribution of employees consists of some problems that result in ineffective and inefficient. The problems include time-consuming for the manual grading process, difficulty in tracking accomplishments, and lack of consistency and objectivity.

i. Time-consuming for the manual grading process

The job of the superiors is not only to perform the day-to-day responsibilities but also to evaluate the employee's performance. Thus, the manual rating process can be demanding as it requires the superiors to spend a lot of time evaluating each employee's performance individually. This is due to the fact that the superiors have to perform a large amount of preparation and documentation before completing the performance rating. This process includes collecting and reviewing information such as performance records and feedback from different sources. As the number of employees within an organization increases, the evaluation of performance also increases. This might delay the progress of the assessment toward completion and increase the superior's workload.

ii. Difficulty in tracking accomplishment

An organization has a wide range of employees with different levels. Each has distinct roles and responsibilities. This diversity has increased the difficulty in implementing a uniform appraisal system to reflect the performance, accomplishment, and growth of employees accurately. In addition, the evaluation criteria for employees mostly include a complex set of metrics. For instance, quality of work, error rate, completion rate, productivity, attendance, and goal achievement. Managing and tracking these various metrics over time requires a system that is able to handle detailed and role-specific metrics. Moreover, if there are any changes in roles or professional development activities, such as shifting the position of an employee or handling new responsibilities, this will complicate the tracking and recording of progress.

iii. Inconsistencies in the manual grading process

The manual rating process may be inherently prone to errors and inconsistencies, which can affect the fairness and reliability of performance assessments. The nature of manual marking is subject to variability due to human error, subjective interpretation, and inconsistent application of evaluation criteria. These issues can lead to discrepancies in assessment results and affect the accuracy and fairness of academic staff appraisals.

The main motivation for developing this project is the limitation of the traditional performance evaluation process. Superior is required to spend a lot of time on preparation and documentation in order to complete the performance rating. Besides, the manual rating process might cause variability. This provides motivation for developing this project, which can automate the evaluation process and ensure fairness and consistency of evaluation results.

1.3 Project Objectives

The main objective of this project is to develop a website to reduce the workload of the superiors while increasing the process of performance evaluation and also to ease the achievements tracking process.

i. To propose an automatic grading system

This website allows academic staff to enter their accomplishments directly into the system. It then automatically grades these accomplishments against preset standards based on the level and role of the academic staff within the university. These standards are designed to reflect the expectations and goals associated with different positions. When the superiors would like to view the performance evaluation reports of academic staff, they can retrieve them easily. This can prevent them from manually grading the academic staff which wastes their time viewing lots of documentation.

ii. To ease the tracking of accomplishments

This website can provide tracking progress over time. The academic staff can update and add their achievements from time to time, ensuring all the information is the latest version, which reflects their ongoing contributions accurately. By maintaining an up-

to-date record, both the academic staff and the superiors can monitor the changes, development, and achievements throughout the year. Moreover, the system supports the function of generalizing report options, which enables the superiors to generate and view comprehensive performance reports within a short time.

iii. Improve the consistency of evaluation

An effective employee appraisal system prioritizes improving data accuracy and consistency to ensure fair and reliable evaluations of academic staff. To achieve this, the system should implement standardized evaluation criteria and scoring rubrics within the same level to provide a uniform framework for performance assessment across all departments. Such standardization is helpful in reducing subjective biases and ensures that all staff within the same level are evaluated against the same benchmarks. Additionally, automating data collection and grading processes can reduce errors associated with manual handling to ensure that evaluations are both accurate and timely. By integrating various data sources such as teaching evaluations, research outputs, and contributions, this system can offer a comprehensive view of an academic's performance, thus enhancing the reliability of the assessments. Furthermore, robust data integrity measures should be put in place to verify the completeness and correctness of the information used in appraisals. These measures collectively contribute to a more objective and consistent appraisal process, fostering fairness and trust among academic staff.

1.4 Project Scope

The scope of the project is to increase the efficiency of the process of evaluating the performance of academic staff by providing a user-friendly website which able to automatically rate the performance against preset criteria based on the user's input. Features include the login, profile management module (academic staff, superiors, and administrator), add, modify, and delete accomplishments, personal participation activities or training, and goal setting, and generating a summarization of performance reports. All the accounts can only be created and added by the administrator.

i. Login module

Administrator, superiors, and academic staff can only access the website by logging

into their specific accounts, which are created by the administrator as the platform is designed to be used for institutional use. Since the administrator, superiors, and academic staff have different privileges. Therefore, distinct functionalities are provided for each role, ensuring they perform specific tasks efficiently.

ii. Profile management module

The superiors and academic staff have their own profile pages on the website. When they first log in to this website, some of the information is required to be filled in. For example, personal information such as contact information, qualifications, and other job-related information. This facilitates the superiors getting to know his or her subordinates. Moreover, the superiors are able to view the achievements of each staff who under their management.

iii. Performance evaluation module

A performance evaluation module is an essential component in this appraisal system which reduces the workload of the superiors. This module is designed to systematically evaluate and record academic staff performance based on preset standards. Moreover, it is aimed at grading academic staff performance that are tailored to different roles and departments to ensure fair and consistent assessments. Only the administrator has permission to modify the evaluation standards to ensure centralized control.

iv. Calendar module

The superiors and academic staff have their own calendars which help them to manage their time effectively. They can add, edit, and delete events in the calendar. Other than that, the superiors have an additional function, which is to schedule meeting time with academic staff. The superiors can first compare their timetables. Then, they can select a time slot that is free for both the superiors and the staff. A confirmation message will be sent to the staff via email. Once the academic staff confirms the appointment, the calendar will update to show the selected time slot as occupied for both the superior's and the staff's calendar.

v. Report generation module

A report with the summarization of academic staff performance can be generated upon the superior's request. These reports provide valuable insights which help the superiors

to make decisions about the staff's compensation and promotions. In addition, reviewing these reports helps them provide advice to those academic staff with weaker performance or incomplete tasks.

vi. Administration module

Administrator can add new academic staff, faculties, and departments. Besides, the administrator can edit the evaluation standards. The administrator can also view all the existing academic staff information who have accounts on the website.

1.5 Impact, significance and contribution

The proposed employee appraisal system is intended to have a great impact on the efficiency and effectiveness of the employees' performance evaluations. The capability of an automatic grading process has benefited the superiors in terms of time and effort. This allows them to focus on other important tasks but ensures that the evaluation is consistently and fairly conducted. The ability to generate reports with a summarization of employees' performance can enhance the decision-making processes such as promotions. In addition, the superiors can directly book a meeting with employees by utilizing the ability of the system to compare the timetables. The employees will receive an email that notifies them to confirm the booking time. This feature decreases the time needed to coordinate meetings and eliminates additional communication through email or phone calls.

Moreover, this website provides a platform for both the superiors and employees to track and view past and current progress and achievements. The user-friendly interface of the system enables employees to regularly update their achievements to ensure that their profile is continuously up to date. The ability to track and view facilitates ongoing communication between superiors and employees, which allows superiors to monitor progress while allowing employees to set goals to address improvements. In addition, the implementation of a standardized employee appraisal system is significant in enhancing the accuracy, fairness, and reliability of performance evaluation. By using uniform evaluation criteria and scoring rubrics within the same level of employees, subjective biases can be avoided, and consistency in assessments can be ensured.

1.6 Summary

The definition and importance of performance appraisal are introduced in this chapter. However, there are some problems faced when using the traditional method of evaluating employee performance. Therefore, establishing a platform to evaluate employee performance is essential to improve the effectiveness of the performance-evaluation process. This project aims to automate the evaluation process while ensuring fairness and consistency of the evaluation result. Modules that will be developed in this project are the login module, profile management module, performance evaluation module, report generation module, calendar module, and administration module.

CHAPTER 2 LITERATURE REVIEW

2.1 Overview

This chapter reviews similar projects and existing websites for employee appraisal systems. The objective of reviewing similar projects is to get an overview of the reason to develop the project and to identify the achievements and limitations of the project, and the improvements that able to make for future work. In addition, the review of websites aims to summarize the strengths and weaknesses for comparison.

2.2 Similar Projects

2.2.1 Online Fuzzy Based Performance Appraisal System [4]

This Online Fuzzy Based Decision Support System for Human Resource PA was proposed with the main objective of automating the employee appraisal process. This can result in reducing manual effort and ensuring a standardized approach to evaluate the academic staff performance. With the implementation of this system, the appraisal process can be streamlined and transparent. The generated performance reports can also be accessed by the superiors and academic staff easily.

In this paper, the author introduces a useful computing tool, an Expert System (ES). ES a decision tools that make use of facts and rules derived from the knowledge get from human experts in a certain field to solve complex problems [5]. The proposed online fuzzy based performance appraisal had integrated some components in ES. Figure 2.2.1.1 shows the architecture of the proposed Fuzzy Based Performance Appraisal System. In this system, decision variables (Figure 2.2.1.2) to determine the academic staff's status will be the input and stored in the Knowledge Base (KB). In the KB, all the necessary information, data rules cases and relationships are stored. In order to provide output (appraisal result), the relevant information and relationships in KB will be searched by an inference engine and lately provide its prediction.

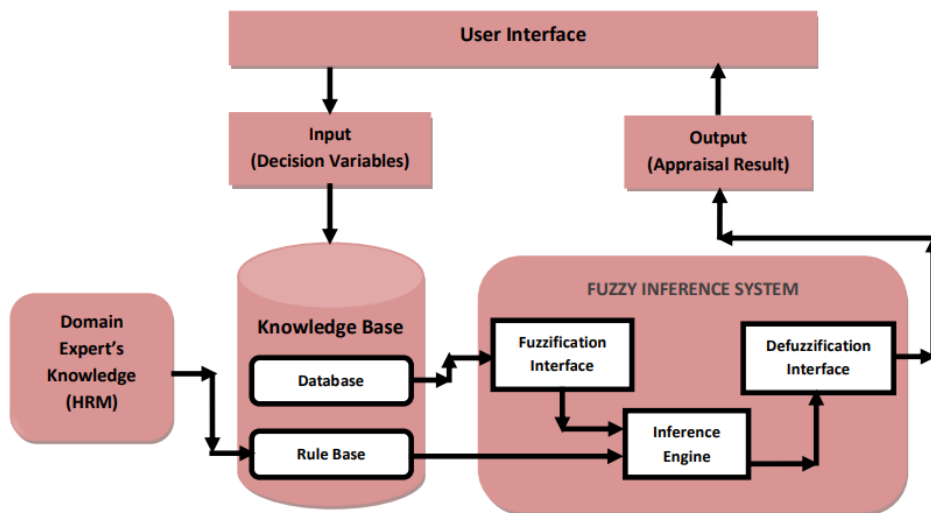


Figure 2.2.1.1 Architecture of the proposed Fuzzy Based Performance Appraisal System [4]



Figure 2.2.1.2 Decision Variables Considered for Appraising Academic Staff Performance [4]

In the stage of experiment, the author had used a combination of HTML, PHP, Asynchronous JavaScript & XML (AJAX), MySQL, and Matrix Laboratory (MATLAB) programming languages to design a work module. By using this module, the information of academic staff can be detected and stored for appraisal purposes. The appraisal system will preprocess the information to a specific format and stored it in the database. The preprocessed data is then loaded into the MATLAB workspace to be accessed by the Fuzzy Inference System. The Fuzzy Inference System will provide an output (performance appraisal result) based on input variables such as academic qualifications and publication number of different categories.

One of the strengths of this fuzzy logic approach is the ability to reduce subjectivity and inconsistency which are often found in common evaluations. By utilizing this fuzzy logic, the system can integrate the main performance evaluation attributes and predicts the appraisal levels of academic staff accurately. In addition, this system provides help for the superiors (HR manager in this case) to handle their tasks better by reducing delays, and external effects on the evaluation results. However, there are some limitations. Due to the fuzzification, rule assessment, and defuzzification process, the implementation is a computationally intensive process that needs a large amount of processing power. Furthermore, the effectiveness of the system is heavily dependent on the accuracy and comprehensiveness of the input data and rules provided, which can be challenging to maintain over time.

2.2.2 Integrated Performance Appraisal System with Management by Objective Method [5]

In this paper, the author developed a performance appraisal system by adopting the Management by Objective (MBO) technique to improve the limitation found using the Behaviourally Anchored Rating Scales (BARS) technique which evaluates performance using the same indicator to evaluate employee performance. This MBO techniques evaluate the performance based on whether each member in the team is successfully hit the target. Thus, this ensures that the evaluation of each employee is unique. In this system, the unit leader is the one who responsibilities to develop a rating scale based on both employee's work behaviour and whether the employee met the standard.

The prototyping methodology is used by the author to develop the system. At first, system analysis is conducted through interviews, observations and review existing documents or reports to find out the problem faced and identify the system requirements. A system design which includes data, process and user interface (UI) is created. This process is iterated until there is one system design is approved. In the system development phase, the authors used PHP programming language for program implementation and Postgre SQL software for data implementation. System testing is carried out in the next phase to ensure that the program can run without error and the output is align with the requirements defined early.

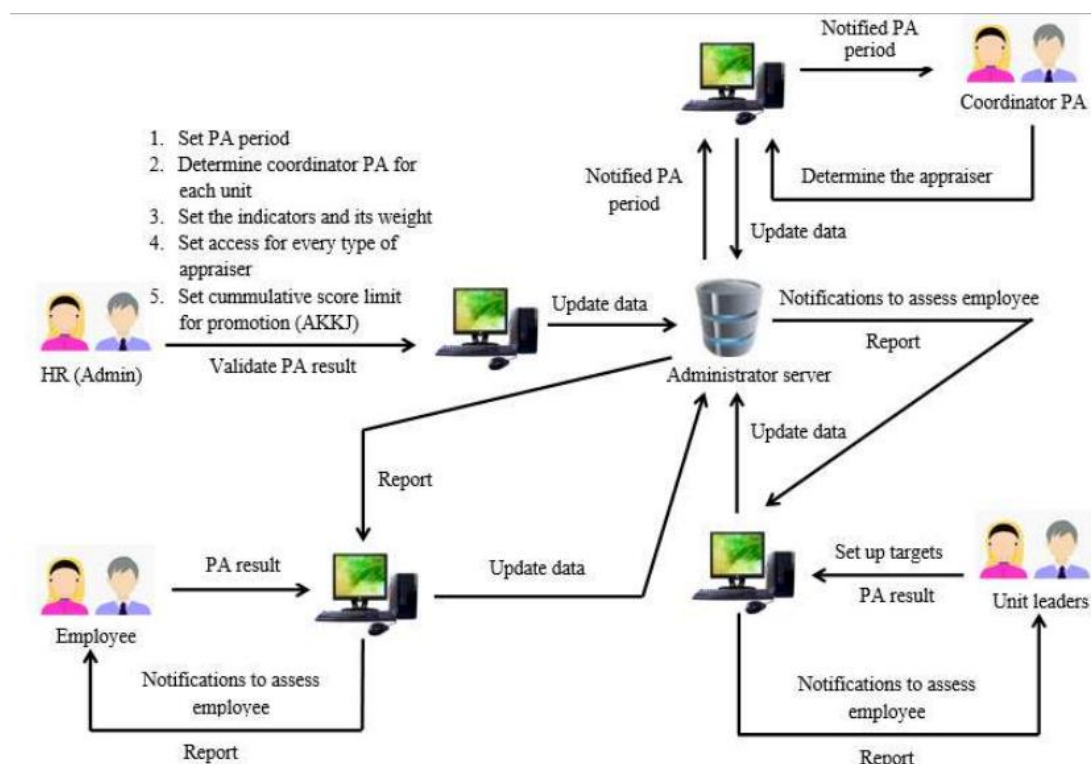


Figure 2.2.2.1 Appraisal System Process using MBO technique [5]

There are five stages of assessment for the proposed appraisal system to work. The Directorate of HR and unit leaders will first set the assessment period, targets, types of appraisers, indicators and weighting of assessments for each type of appraiser. For each assessment period, the unit leader will be determined the work target of employee. Prospective employee appraisers are then identified in the second stage. Next, in the assessment stage, both the unit leader and employees are notified to provide evaluation on themselves and colleagues. A report will be generated in the fourth stage after the

CHAPTER 2 LITERATURE REVIEW

performance value of each employee is calculated. Lastly, the HR Directorate will validate the report generated.

The use of the MBO approach make the system to be more effective in evaluating employee performance. This is due to the reason that the evaluation is based on the actual work outcomes rather than just the employee behavior, where the system can provide a more accurate reflection of an employee's contributions. This can ensure that the evaluation is closely aligned with specific duties and responsibilities assigned to each employee. However, the complexity and time needed to set up the system is a limitation. This is due to the fact that it requires the formulation of particular work objectives and assessment criteria for each individual employee, which can be a time-consuming process.

2.3 Existing Websites

2.3.1 Employee Performance Evaluation System in PHP/MySQLi [6]

This Employee Performance Evaluation System is a simple project that was created to assist companies in evaluating the performance of employees according to their task accomplishments. The system supports three user types, which is administrator, evaluator, and employee.

All three types of users can log in to their specific account by selecting the user types (Figure 2.3.1.1). For administrators, after they successfully log in to the website, they can see various categories such as tasks, evaluation, departments, designations, employees, evaluator, and users on their dashboard. For evaluator, there are two categories which are tasks and evaluation while for the employees, they can see the tasks category.

For the admin's task tab, they can view all existing tasks with each due date, who had been assigned to the task, and the status and action that can be taken on each task. The actions that can be taken on the existing task are to edit or delete the task, to view the task that shows more detailed information such as the task description, and to view the progress of the task. The admin can also add new tasks. Moreover, if there is new department or designation created within the company, the admin can add them into the system. The admin can also add new evaluators or new employees.

On the other hand, the evaluator and employee can log in to their account that was created by the admin. For the evaluator, they can view the existing tasks and the progress of the task from the task tab. For the evaluation tab, the evaluator can select the employee and the task under the selected employee they would like to rate the performance. After the completion of the selection, the progress of the task that updated by the employee will appear. This provides convenience for the evaluator to rate the performance of the employee in terms of efficiency. Before submitting the evaluation, the evaluator can provide some comments in the remarks section. For the employee, they are only allowed to view the tasks that are assigned to them with some information such as the due date and the description of tasks. Actions can be taken on each task are view task, view progress, and add progress.

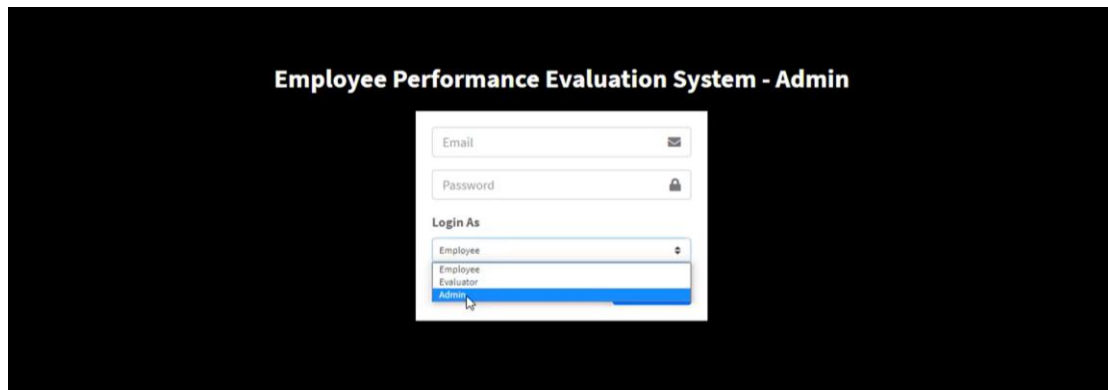


Figure 2.3.1.1 Log in page [6]

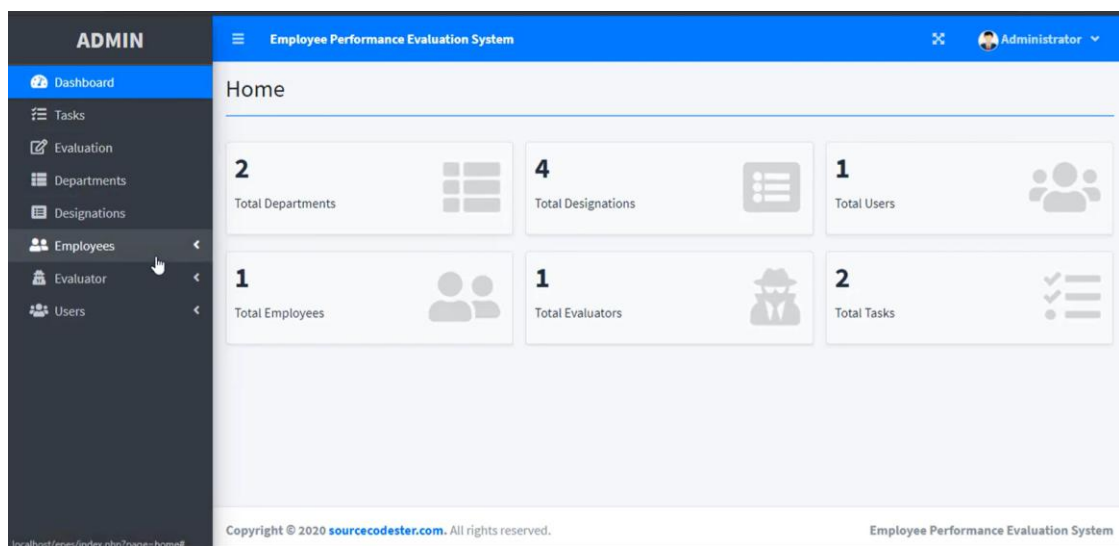


Figure 2.3.1.2 Admin's main page [6]

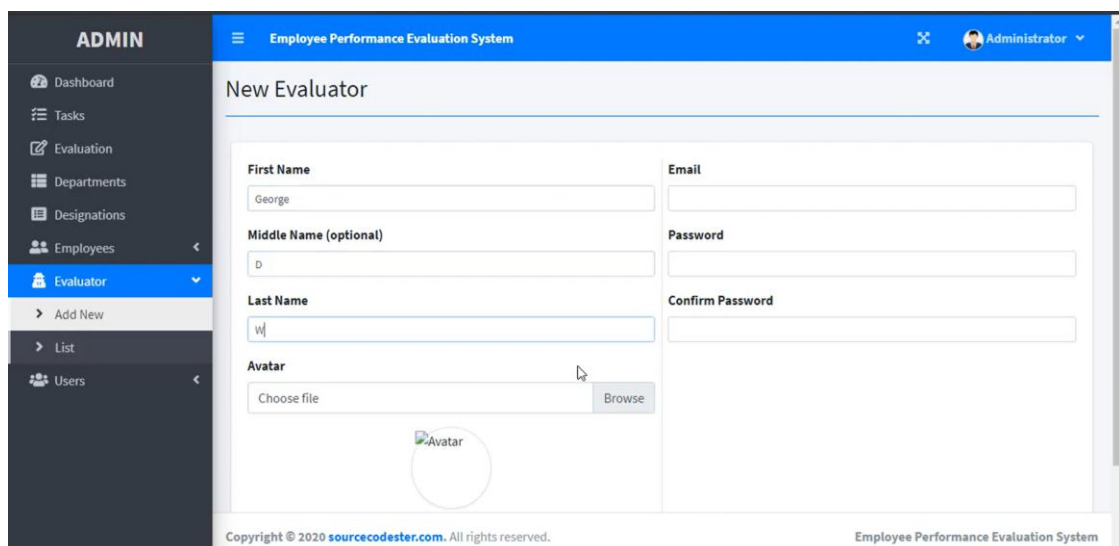


Figure 2.3.1.3 Admin's add evaluator page [6]

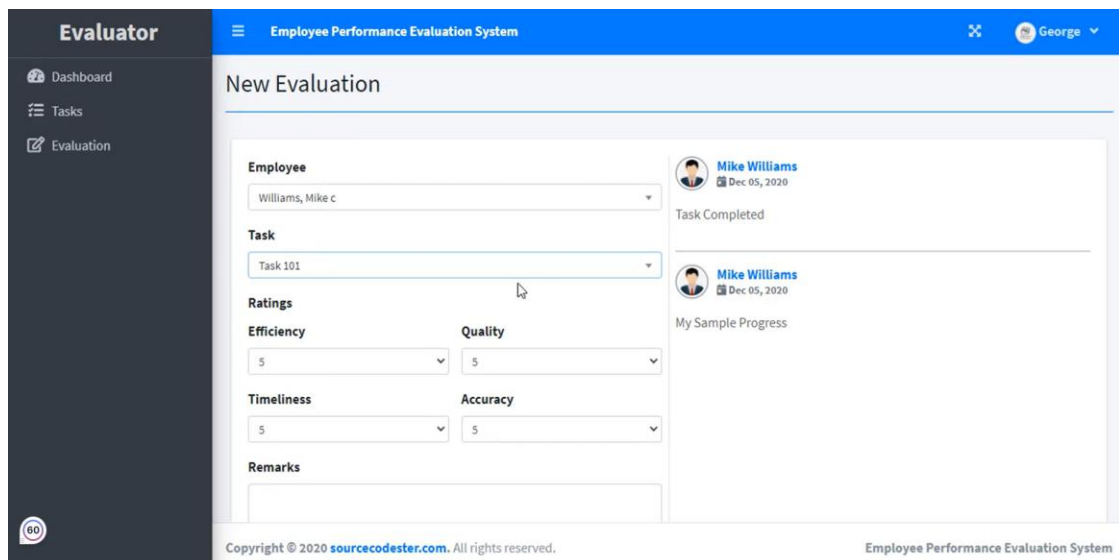


Figure 2.3.1.4 Evaluator's (superior) evaluation page [6]

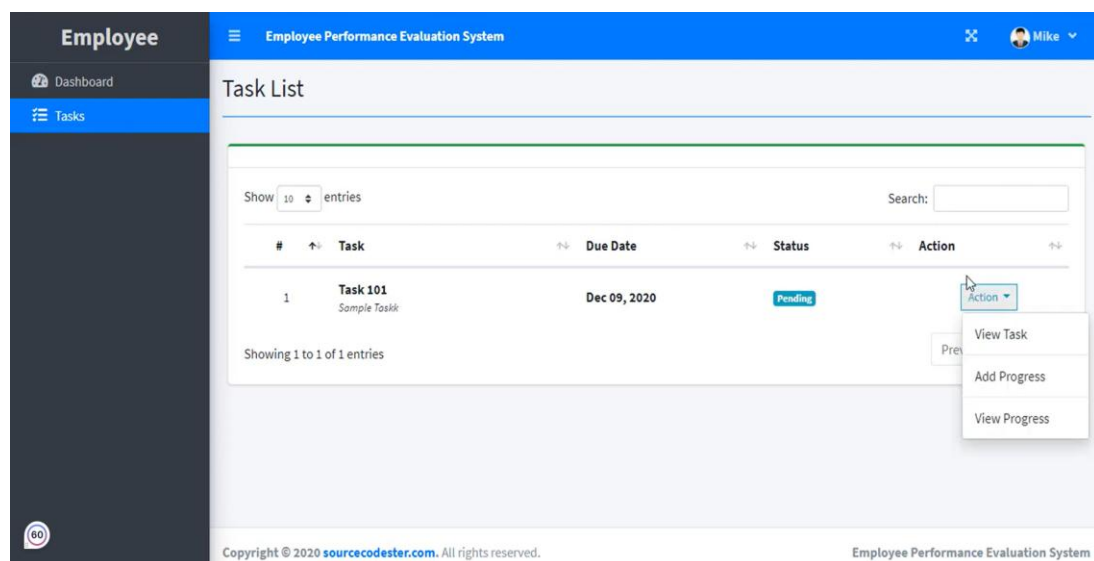


Figure 2.3.1.5 Employee's task list page [6]

Strength

- Only the admin is permitted to add new employees and evaluators.
- The admin can assign tasks to different employees.
- The evaluator can rate the employee's performance on specific tasks assigned by the admin.
- The evaluator can view all the existing tasks from the task list and view the progress of each task.

- The employee can view the task assigned by the admin with detailed information such as the task description and due date.
- The employee can update their progress and mark the task as completed.

Weakness

- The website only supports manual evaluation of employee performance.
- The evaluation of employee performance is based on each task they complete.

2.3.2 PeopleHR's performance appraisal system [7]

PeopleHR's performance appraisal system is created to enhance the way organizations manage and evaluate the employee's performance.

For managers, the main page after they log in to their account shows the total number of employees who report under them and the status of each team member (Figure 2.3.2.1). The manager can also assign tasks to employees who report under them through the tasks tab as shown in Figure 2.3.2.2. In addition, the manager can view all the tasks assigned to them which had been categorized. For example, overdue tasks, new tasks, and completed tasks. Through the 'team' tab, the manager can rate the employees and print the evaluation reports after completing the evaluation. Moreover, the manager can view the evaluation done by employees themselves if any (Figure 2.3.2.3).

For employees, they can update their status on the calendar provided under the 'planner' section. For example, if the employee is absent due to the reason of sickness, they can select a sad face which indicates they had applied for a leave. For the 'Me' tab, the employee can rate their performance. This evaluation can be accessed by the manager if it is completed. Moreover, the employees are able to set their goals to motivate themselves and to prioritize the work.

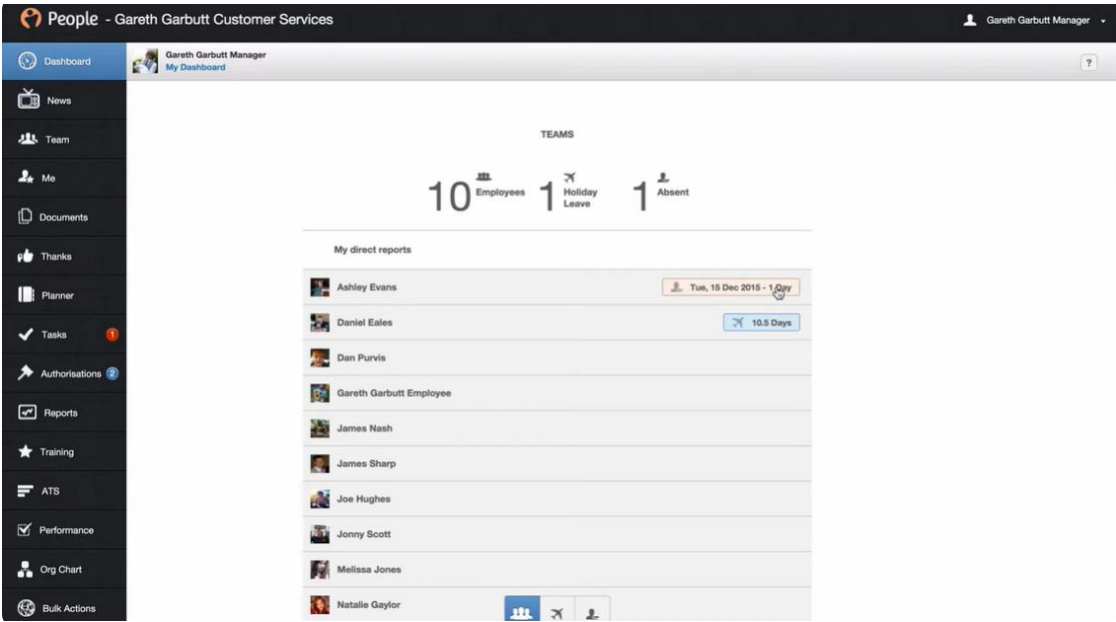


Figure 2.3.2.1 Manager’s main page [7]

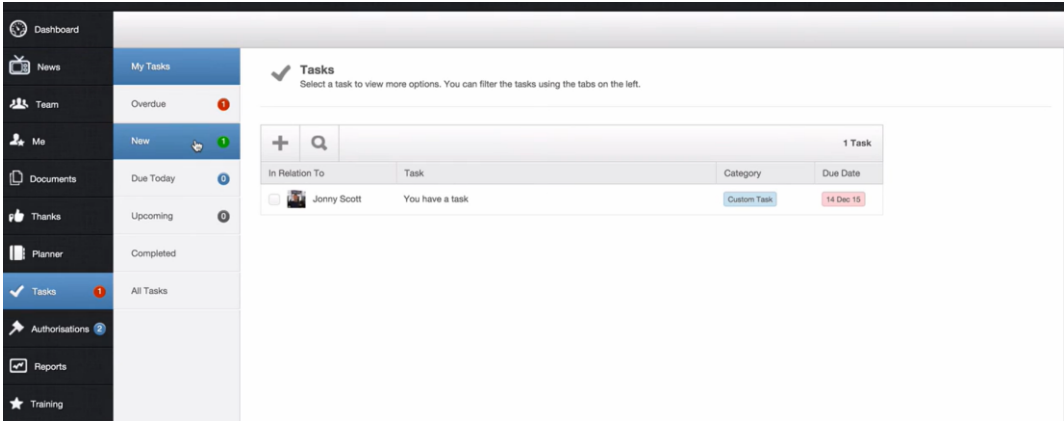


Figure 2.3.2.2 Manager’s tasks page [7]

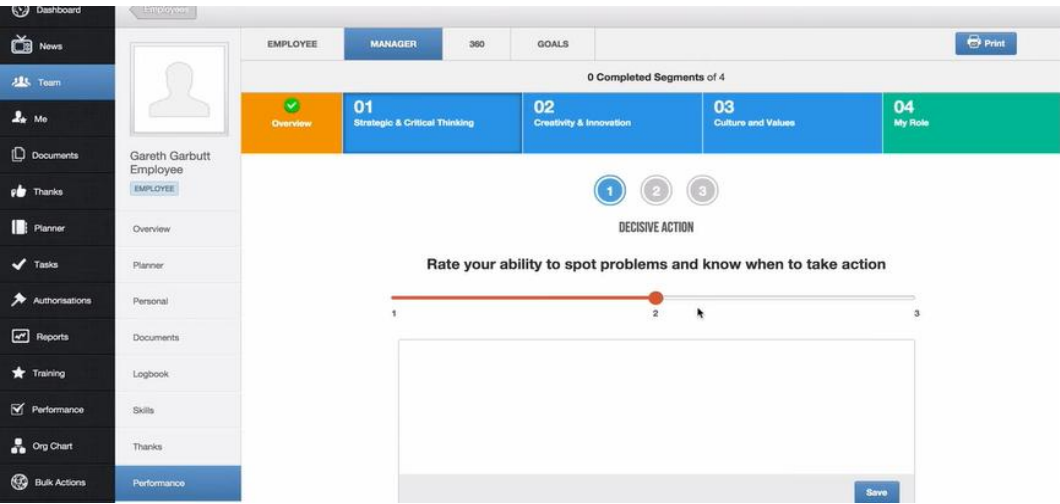


Figure 2.3.2.3 Manager’s evaluation page [7]

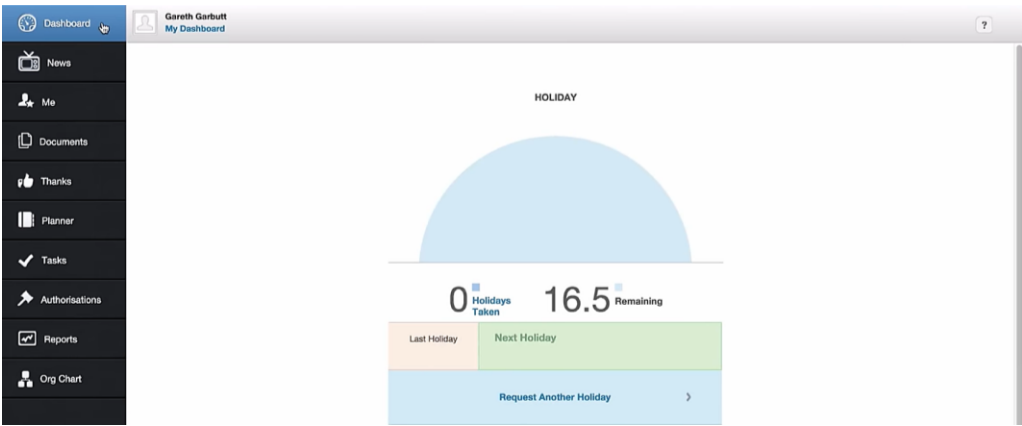


Figure 2.3.2.4 Employee’s main page [7]

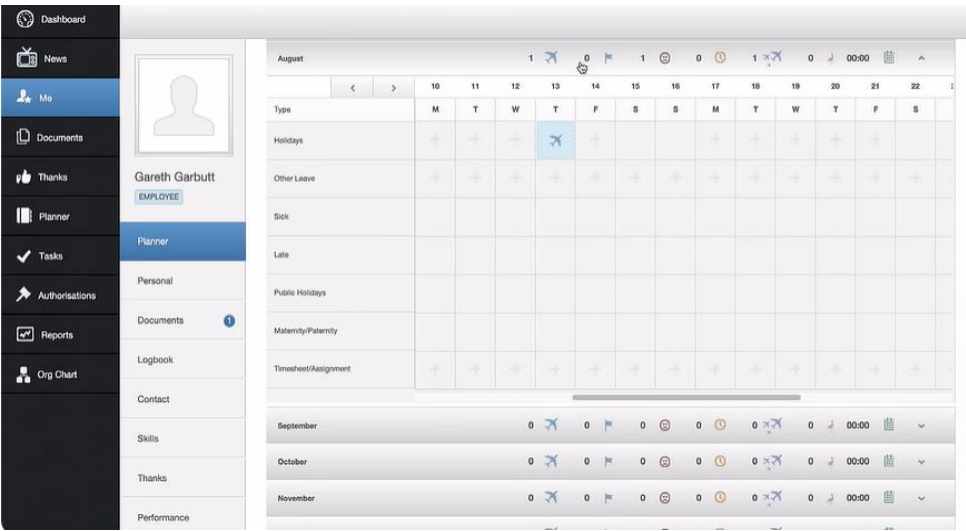


Figure 2.3.2.5 Employee’s planner page [7]

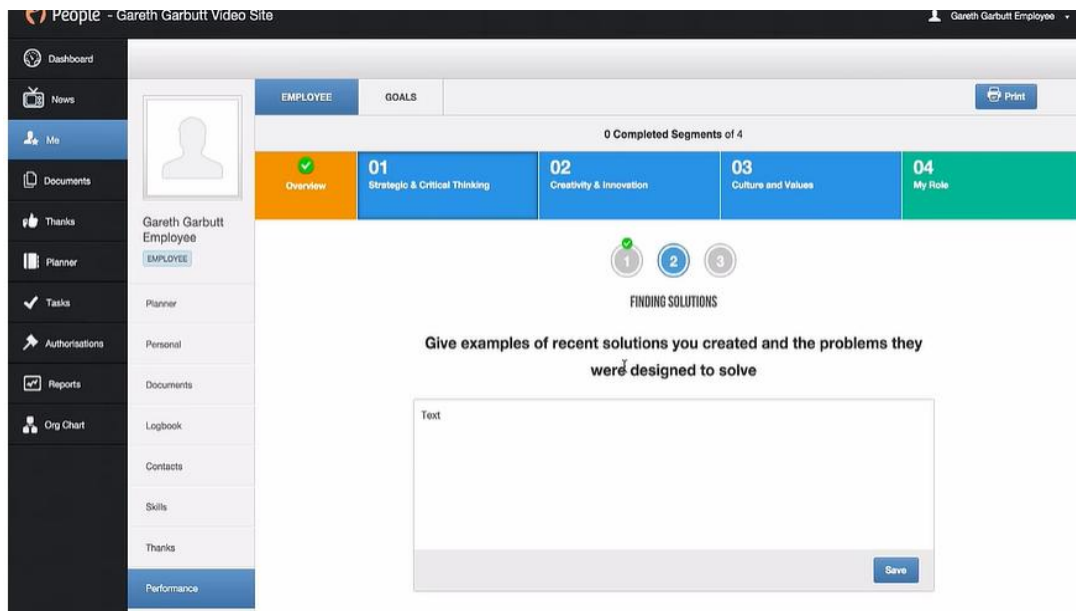


Figure 2.3.2.6 Employee's evaluation page [7]

Strength

- The manager and employee can track all the past reviews.
- The platform supports the news function which makes it easy for all the users to receive announcements.
- The manager can view the employees' planners to see whether they are on duty or on leave.
- There is an 'Organizational Chart' tab for users to view the reporting or the relationship hierarchy.

Weakness

- The employee can evaluate their performance which may cause the evaluation result to be not trustable.

2.3.3 Trakstar – Employee Performance Appraisal System [8]

Trackstar is a software company founded in 2001 that focuses on creating easy-to-use software such as performance reviews and candidate sourcing. This employee performance appraisal system is created with 3 main functions which are scoring the appraisal, setting goals, and running reports.

CHAPTER 2 LITERATURE REVIEW

First, for scoring the appraisal, after the manager successfully logs in to their account, they are provided with a few tabs such as task list, appraisals, track goals, and reports. For the task list page, the manager can select employees in order to score their appraisal. Other than rating the employees' performance, the manager can also view the comments either from other employees or the employee (if any) under the core value section. The manager can also add new comments to support their rating. Next, is to set goals. Both managers and employees (optionally) can set goals that aims to quantify competencies by adding measurements.

Lastly, this system supports the administrator to run the existing reports. The administrator can search for past appraisals by entering the name, for example, 2012 Appraisal, and apply some filters for more precise results. The results generated can be exported as an Excel file upon the administrator's request. Moreover, the administrator can view the selected report in four categories, which are rater bias, performance history, rating distribution, and performance ranking. For a smaller scale, the administrator can select to view different departments' reports and different groupings.

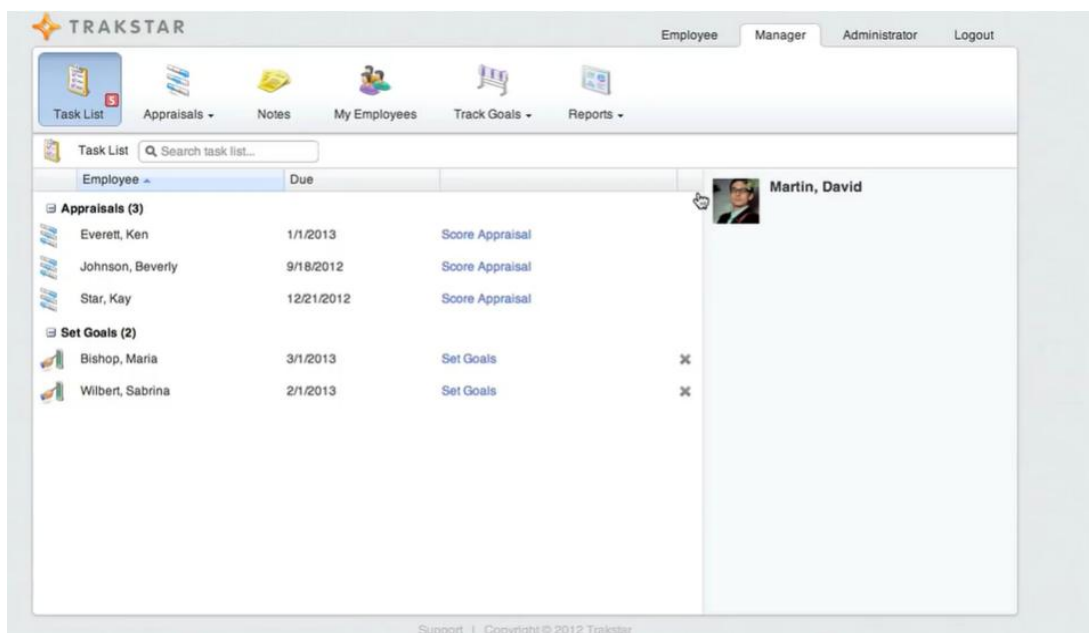


Figure 2.3.3.1 Manager's tasks list page [8]

CHAPTER 2 LITERATURE REVIEW

Task List Appraisals Notes My Employees Track Goals Reports

Appraisal Workshop Score Appraisal

Quarterly Appraisal for Kay Star

Position Description: Support [Manage Goals and Notes](#)

[Return to Workshop](#)

Core Values Section

Core Values for all Dundermifflin employees 25%

Work Quality

Able to achieve desired outcomes with a minimum of avoidable errors and problems. 50%

Not Effective Minimally Effective Effective Highly Effective Exceptional

Notes (2)

David Martin 3 minutes ago
On 12/1 a customer sent you a great kudos saying "Fantastic job managing our account!"
Great work, Kay. Keep it up!
[Copy to Comment](#)

Kay Star about a minute ago

Performance Appraisal	Rating
Core Values Section	-
Work Quality	-
Analytical Thinking	-
Management Section	-
Approachability	-
Objective 2	-
Revenue	-
Areas for Improvement	-
Strengths	-
Summary Comments	-
Overall Score	-

Period: 10/1/2012 - 12/21/2012
Appraisal Due Date: 12/21/2012
Manager: David Martin [more](#)

Figure 2.3.3.2 Manager's evaluation page for selected employees [8]

Task List Employees Appraisals Processes Positions Competencies Groups Reports Settings

Appraisals Search Appraisals

Employee	Manager	Position	Due Date	Emp	Mgr	Apr
Antler, Randy	Everett, Ken	Staff Engineer	10/15/2012	🔒	🔒	
Clark, Heather	Dowd, Joe	Human Resource Spec...	4/1/2011	✅	✅	
Everett, Ken	Martin, David	VP Engineering	1/1/2013	🔒	🔒	
Fooser, Harry	Everett, Ken	Staff Engineer	10/15/2012	🔒	🔒	
Johnson, Beverly	Martin, David	Director of Finance	9/18/2012	🔒	🔒	
Kunce, Renee	Wilbert, Sabrina	Support Representative	11/15/2012	🔒	🔒	
Lately, Johnny	Mishap, Mark	Staff	1/1/2013	🔒	🔒	
Maback, Pat	Mishap, Mark	Staff	1/1/2013	🔒	🔒	
Martin, Glenn	Bishop, Maria	Warehouser	1/1/2013	🔒	🔒	
Rieken, Julie	Wilbert, Sabrina	Sales Executive	9/18/2012	🔒	🔒	
Star, Kay	Martin, David	Support	12/21/2012	✅	✅	
Talcott, Mack	Everett, Ken	Staff Engineer	1/1/2013	🔒	🔒	
ToGo, Jonesin	Mishap, Mark	Staff	1/1/2013	🔒	🔒	
Ussly, Siri	Mishap, Mark	Staff	1/1/2013	🔒	🔒	
Vorwallier, Marcus	Everett, Ken	Staff Engineer	10/1/2012	🔒	🔒	

Draft
Goals and Notes
Scoring (16)
Archived

Figure 2.3.3.3 Administrator's appraisal page [8]

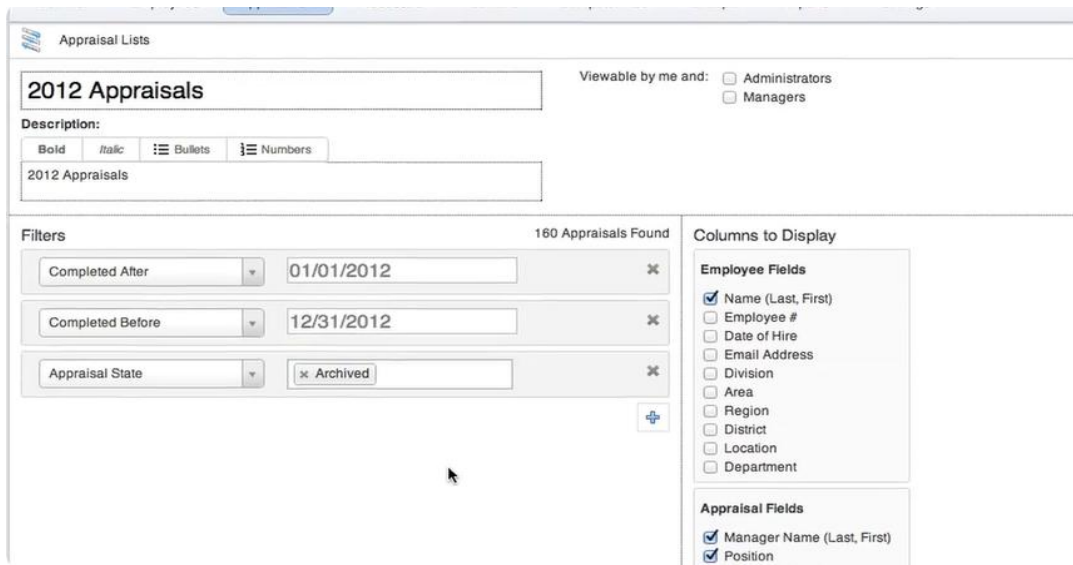


Figure 2.3.3.4 Administrator's appraisal page (Search for certain year appraisal list)
[8]

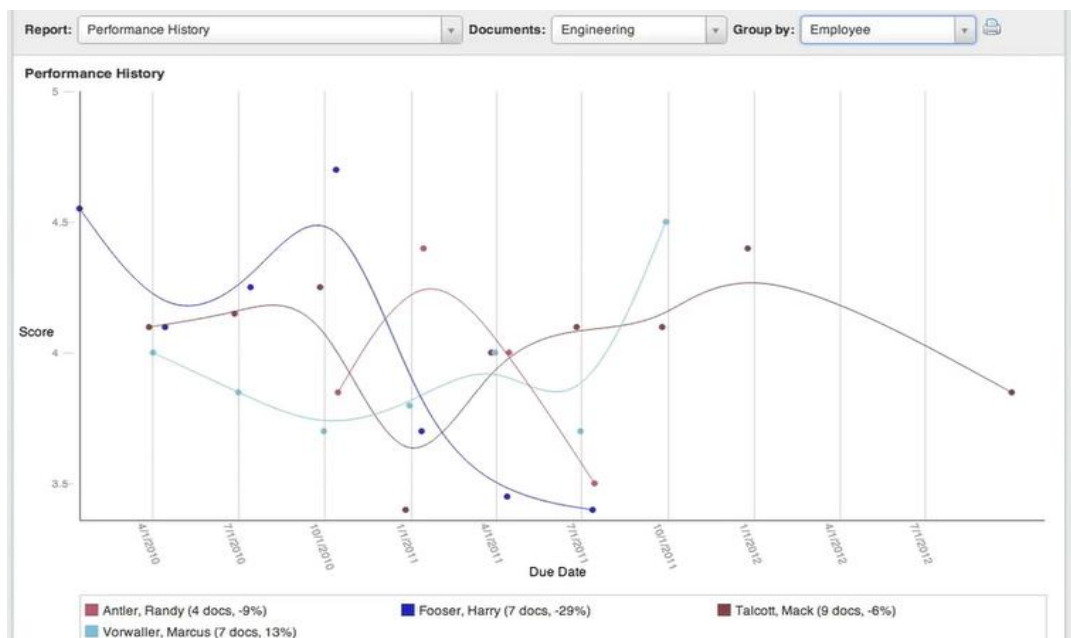


Figure 2.3.3.5 Administrator's performance history page for selected department [8]

Strength

- The administrator can view a clearer version of the employees' past performance in graph form.
- The administrator can track the past appraisal reviews.
- The manager can rate the performance of employees and leave some

comments under each evaluation criteria.

Weakness

- The system only supports the manual rating process.
- The schedule for employees is not provided, which is not convenient for the manager to schedule meetings with the selected employee.

2.4 Summary

In this chapter, two similar projects with distinct approaches have been reviewed. The first project is the Online Fuzzy Based Performance Appraisal System. This system seeks to automate the appraisal process for academic staff by integrating a Fuzzy Inference System within an ES framework. The objective is to reduce the manual efforts and to improve standardization and transparency in the process of performance evaluation. The system uses various programming technologies, including HTML, PHP, AJAX, MySQL, and MATLAB to collect and process data. Then, the Fuzzy Inference System evaluates the staff's performance based on the input variables such as academic qualification publication records. Lastly, the performance reports are generated which are accessible to both superiors and staff. On the other hand, the Integrated Performance Appraisal System with Management by Objective Method focuses on improving the traditional BARS technique by implementing the MBO method. This system evaluates employee performance based on their ability to meet set targets, providing a unique assessment for each team member. The development process involved prototyping, where system requirements were gathered through interviews and document reviews. The final system, built using PHP and PostgreSQL, includes stages for setting targets, identifying appraisers, conducting evaluations, and generating and validating performance reports. The MBO method ensures that evaluations are closely aligned with individual targets, offering a more tailored appraisal approach.

Three existing websites, including Employee Performance Evaluation System in PHP/MySQLi, PeopleHR's performance appraisal system, and Trakstar – Employee Performance Appraisal System are reviewed. Their strengths and weaknesses are then listed out and concluded in Table 2.4.1.

Table 2.4.1 Comparison of strengths and weaknesses of existing websites

	Employee Performance Evaluation System in PHP/MySQLi	PeopleHR's performance appraisal system	Trakstar – Employee Performance Appraisal System
Design of User Interface	Simple design.	Simple design.	Simple design but packed for the appraisal section.
Type of Roles	Administrator, evaluator, employee	Administrator, manager, employee	Administrator, manager, employee
Type of Rating Process	Manual rating process.	Manual rating process.	Manual rating process.
Responsibilities of Assigning Task	Administrator.	Manager.	Manager.
Information of Manager/Evaluator	Not provided.	An organizational chart is provided in this system with brief information.	Not provided.
Graph-represented of reports	Not provided.	Not provided.	Graph-represented reports are supported in this system.
Announcement	Not provided.	A News section is provided to add new announcements.	Not provided.
Event Calendar	Not provided.	Provided for both manager and employee to update their events.	Not provided.

CHAPTER 3 PROPOSED METHOD/APPROACH

3.1 Overview

In this chapter, the technologies and methodology used to develop the project is proposed. The system requirements including the hardware and software specifications, and the steps of methodology will be explained.

3.2 Design Specifications

3.2.1 Methodologies and General Work Procedures

Rapid Application Development (RAD) methodology is the methodology proposed for the project. RAD methodology is a software development methodology that focuses on the quick development of a program. This methodology allows the project to perform various iterations and improvements during software development.

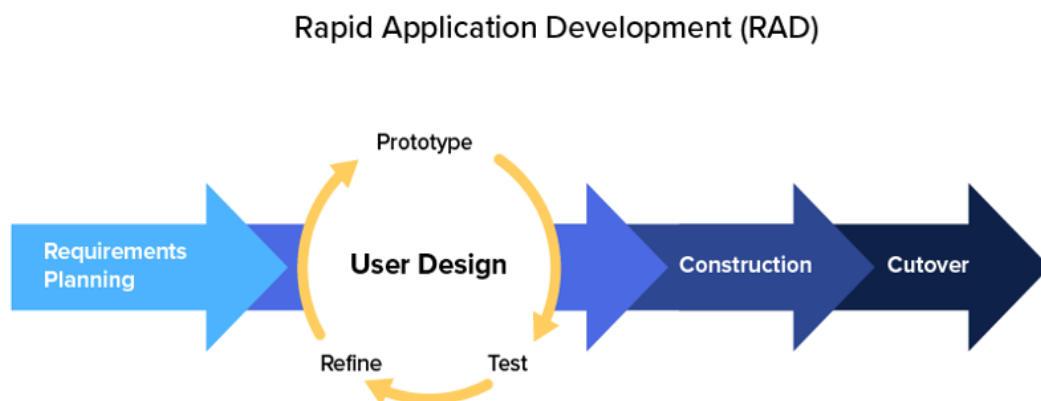


Figure 3.2.1.1 RAD Methodology [9]

3.2.1.1 Requirements Planning Phase

The first phase of RAD methodology is the requirements planning phase. Following the identification of problem statements, a proposal was made for the development of

a comprehensive employee appraisal system. Furthermore, the objective of the project will be listed during the planning and analysis phase. A review of existing literature and systems will be conducted to determine the necessary requirements for users of the proposed website. For example, the Employee Performance Evaluation System in PHP/MySQLi, PeopleHR's performance appraisal system, and Trakstar – Employee Performance Appraisal System will be conducted to identify their strengths and weaknesses. The objective is to develop a well-structured website and to define the project requirements. In addition, the system specification (hardware and software) and the project timeline will be established in this phase.

3.2.1.2 User Design Phase

The next phase is the user design phase. In this phase, the prototype will be created that includes all the modules such as the login module, profile module, performance evaluation module, calendar module, performance report generation module, and administration module. The languages and technologies that will be used in developing the functionalities of the website are MongoDB, Node.js, Express.js, JavaScript, while React is used alongside with HTML and Tailwind CSS to design the user interface. After the initial version of the program is created, it will undergo a testing phase, and the prototype will be iterated if necessary for improvements or newly defined requirements. Therefore, the prototype will include only functions and basic UI design. This phase will be carried out again and again before the project owners acknowledge the finalization of the prototype's function.

3.2.1.3 Construction Phase

In the construction phase of RAD methodology, it mainly focuses on building and refining the system. The system will first break down into small modules where each module represents a function of the system and coding is written. All the modules will be integrated into a system, and testing will be carried out to ensure all the modules work well together. HTML, Tailwind CSS will be used to improve the user interface design. Lastly, the user's feedback is gathered to make alterations or include new functionality until the expectation of users is met.

3.2.1.4 Cutover Phase

This is the final phase of RAD methodology where final testing for all modules is conducted to ensure that the system (website) performs as expected without any issues or errors. This includes checking system functionality and usability. Some of the documentation such as system design, operation, and testing results needs to be completed after the completion of all phases. After all the documentation is done, the project has now reached completion and is ready to be launched.

3.2.2 Tools to Use

i. Hardware Requirements

Table 3.2.2.1 The hardware component and requirements for website development

Component	Requirements
Windows	Windows 11 Home Single Language
Processor	Intel® Core™ i5-1135G7 @ 2.40GHz
Graphic Card	Intel® Iris® Xe Graphics
RAM	8.00 GB (7.80 GB usable)
Input	Keyboard and mouse
System Type	64-bit operating system, x64-based processor

ii. Software Requirements

Table 3.2.2.2 The software component and requirements for website development

Component	Requirements
Tools	<p><u>Visual Studio Code</u></p> <p>Visual Studio Code is a free, open-source code editor developed by Microsoft. It supports various programming languages and offers a rich set of features, including debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and version control integration.</p>
	<p><u>MongoDB</u></p> <p>MongoDB is an open-source NoSQL database system. It stores data in JSON-like format called BSON which makes it ideal for handling non-relational attributes.</p>
	<p><u>Node.js</u></p> <p>Node.js is an open-source JavaScript runtime environment that supports cross-platform functions. It allows JavaScript code to run outside of the browser, usually on the server side.</p>
Languages, Libraries and Frameworks	<p><u>Hypertext Markup Language (HTML)</u></p> <p>HTML is the standard markup language used to create and design web pages and web applications. It structures content on the web by defining elements such as headings, paragraphs, links, images, and other content.</p>
	<p><u>Tailwind CSS</u></p> <p>Tailwind CSS is a utility-first framework to build custom UI quickly by using reusable utility classes without writing custom CSS.</p>
	<p><u>JavaScript</u></p> <p>JavaScript is a high-level, interpreted scripting language that enables dynamic content on web pages. It is essential for client-side scripting and allows for interactive features such as form validation, animations, and asynchronous content loading.</p>
	<p><u>React</u></p> <p>React is an open-source JavaScript library used to build user interfaces (UI), primarily for single-page applications which allow the creation of dynamic web components.</p>

	<p><u>Express</u></p> <p>Express is a web application framework for Node.js that has been developed for the purpose of simplifying the process of building and managing web applications and APIs by offering tools to handle routing, middleware, HTTP requests and responses.</p>
--	--

3.3 System Design/Overview

3.3.1 Use-Case Diagram

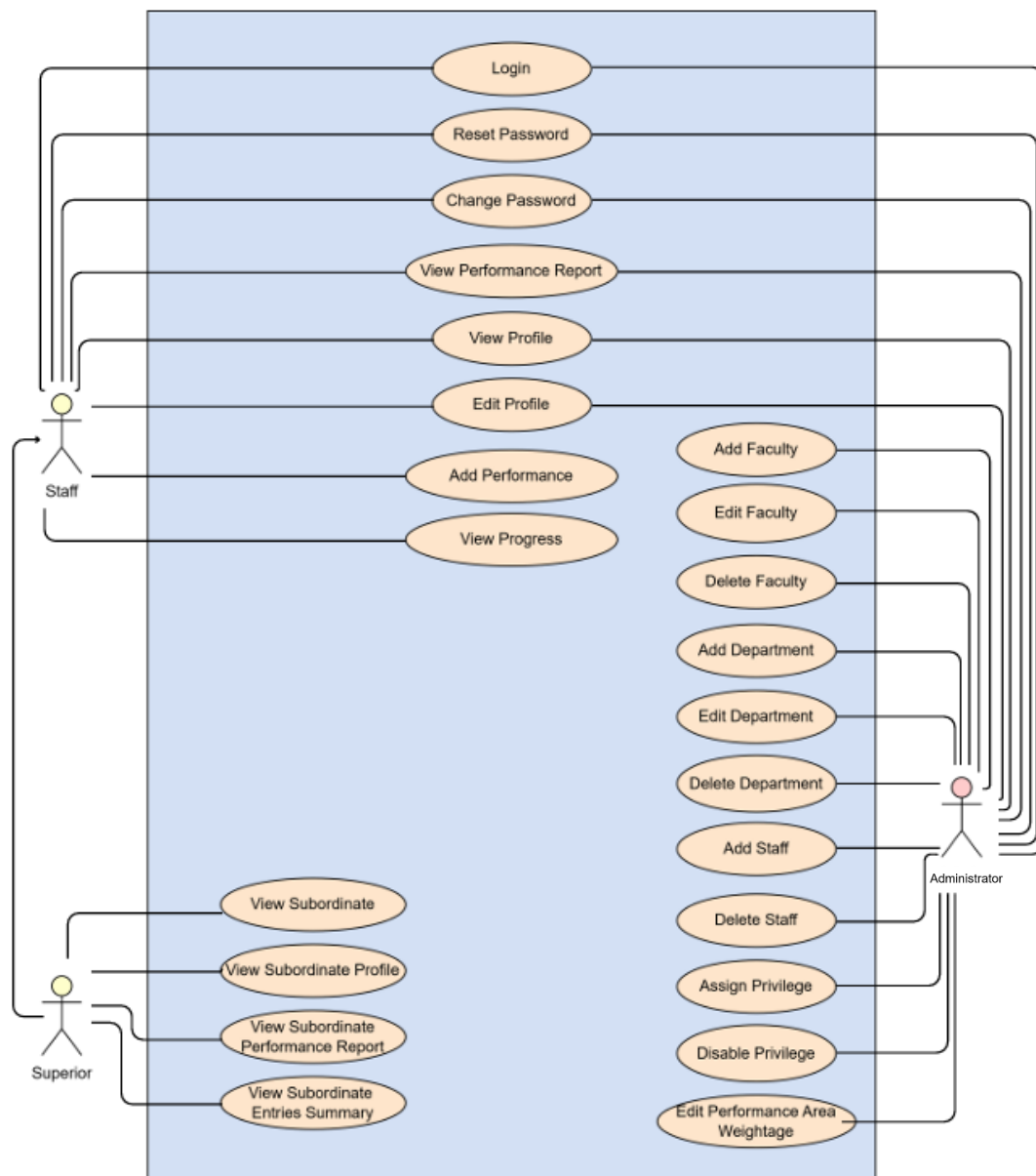


Figure 3.3.1.1 Use-case Diagram for Comprehensive Employee Appraisal System

3.3.2 Use-Cases Description

Table 3.3.2.1 Login Use Case Description

Use Case ID	00001
Use Case Name	Login
Brief Description	Staff / Superior / Administrator are allowed to access the system.
Actor	Staff, Superior, Administrator
Trigger	Click the login button in the home page.
Precondition	The account must exist in the system and have not logged in.
Normal flow of events	<ol style="list-style-type: none"> 1. Staff / Superior / Administrator fill in the email and password. 2. The system checks whether both email and password are provided. 3. The system searches for matching accounts using the email provided and compares the provided password with the stored hashed password. 4. The system generates a JWT token. 5. The system sets the JWT token in a secure HTTP-only cookie (valid for 7 days). 6. The system redirects the user (staff / superior / administrator) to their dashboard.
Alternative flow	<p>5a. Staff / Superior / Administrator do not need to log in again within 7 days.</p> <p>5b. After the token expires after 7 days, the user needs to log in again.</p>
Exception flow	<p>2a. If email or password is not provided, an error message shows. Staff / Superior / Administrator must fill in both fields.</p> <p>3a. If the provided email does not exist or the provided password does not match the stored hashed password, an error message shows. Staff / Superior / Administrator must enter a valid email and re-enter the correct password.</p>

Table 3.3.2.2 Reset Password Use Case Description

Use Case ID	00002
Use Case Name	Reset Password
Brief Description	Staff/ Superior / Administrator are allowed to reset password if they forget their password.
Actor	Staff, Superior, Administrator
Trigger	Click the forget password button in the login page.
Precondition	The account must exist in the system and staff / superior / administrator forgets their password.
Normal flow of events	<ol style="list-style-type: none"> 1. Staff/ Superior / Administrator enter and submit their email, the received OTP, and a new password. 2. The system verifies that all three fields (email, OTP, new password) are provided. 3. The system searches for a user account with the email provided. 4. The system checks if the provided OTP matches the one saved in the user's record. 5. The system verifies that the OTP has not expired (within 15 minutes). 6. The system hashes the new password and updates it in the database. 7. The system clears the OTP and expiry time from the user's record. 8. The system returns a success message: "Password reset successfully".
Alternative flow	-
Exception flow	<ol style="list-style-type: none"> 2a. The system returns an error message if any of the required fields (email, OTP, or new password) is not filled in. 3a. If the user account does not exist, an error message returns. 4a. If OTP is not provided or is incorrect, an error message returns. 5a. If the OTP has expired, an error message returns.

Table 3.3.2.3 Change Password Use Case Description

Use Case ID	00003
Use Case Name	Change Password
Brief Description	Staff / Superior / Administrator are allowed to change their account

CHAPTER 3 PROPOSED METHOD/APPROACH

	password.
Actor	Staff, Superior, Administrator
Trigger	Click the change password button for administrator or the profile button at the sidebar for staff / superior.
Precondition	Staff / Superior / Administrator must be authenticated and logged in.
Normal flow of events	<ol style="list-style-type: none"> 1. Staff / Superior / Administrator enter and submit the current password and the new password. 2. The system verifies that both fields (current password and new password) are provided. 3. The system checks whether the user exists and whether the current password provided matches the stored hashed password. 4. The system updates the hashed new password.
Alternative flow	-
Exception flow	<ol style="list-style-type: none"> 2a. The system returns an error message if any of the required fields (current password and new password) are not filled in. 3a. The system returns an error message if the user account is not found or if the current password provided does not match the stored hashed password.

Table 3.3.2.4 View Profile Use Case Description

Use Case ID	00004
Use Case Name	View Profile
Brief Description	The profile will show the personal and job-related information of the staff / superior.
Actor	Staff, Superior
Trigger	Click the profile button from the side navigation bar.
Precondition	Staff / Superior must be logged in.
Normal flow of events	<ol style="list-style-type: none"> 1. Superior can view their personal and job-related details like name, email address, faculty, department, phone number, designation and profile picture. 2. Superior can also change their password in this page.
Alternative flow	-
Exception flow	-

Table 3.3.2.5 Edit Profile Use Case Description

Use Case ID	00005
Use Case Name	Edit Profile
Brief Description	Staff / Superior can edit certain information like profile picture, name, phone number, qualification, and areas of expertise.
Actor	Staff, Superior
Trigger	Re-enter the information for the fields that want to have modification.
Precondition	Staff / Superior must be logged in and be in the profile page.
Normal flow of events	<ol style="list-style-type: none"> 1. Staff / Superior clicks the profile button from the side navigation bar. 2. Staff / Superior re-enters information for fields that would like to have some changes. 3. Latest information is updated and saved after the “Update Profile” button is clicked.
Alternative flow	-
Exception flow	-

Table 3.3.2.6 Add Performance Use Case Description

Use Case ID	00006
Use Case Name	Add Performance
Brief Description	Staff / Superior can add a new performance entry for different areas.
Actor	Staff, Superior
Trigger	Select the performance area and click on the add entry button.
Precondition	Staff / Superior must be logged in and be in the performance area page.
Normal flow of events	<ol style="list-style-type: none"> 1. Staff / Superior clicks on the performance area button from the side navigation bar. 2. Staff / Superior selects the performance area they would like to add a new entry. 3. Staff / Superior clicks on the add entry button. 4. A form modal will pop out which requires the staff / superior to fill in the information needed. 5. Staff / Superior clicks on the submit button after all fields are filled in.

CHAPTER 3 PROPOSED METHOD/APPROACH

Alternative flow	-
Exception flow	-

Table 3.3.2.7 View Performance Report Use Case Description (Staff / Superior)

Use Case ID	00007
Use Case Name	View Performance Report
Brief Description	Staff / Superior can view the performance report that contains the summary of performance and the progress of each area.
Actor	Staff, Superior
Trigger	Click on the performance reporting button from the side navigation bar.
Precondition	Staff / Superior must be logged in.
Normal flow of events	<ol style="list-style-type: none"> 1. Staff / Superior clicks on the performance reporting button from the side navigation bar. 2. The staff / superior performance is displayed. 3. Staff / Superior can view the summary of all the achievements added by themselves, and the progress of each entry in a progress bar and bar chart format.
Alternative flow	-
Exception flow	-

Table 3.3.2.8 View Performance Report Use Case Description (Administrator)

Use Case ID	00008
Use Case Name	View Performance Report
Brief Description	Administrator can view all the staff's performance reports.
Actor	Administrator
Trigger	Click on the view report button on the profile of the selected staff.
Precondition	Administrator must be logged in and be in the academic staff page.
Normal flow of events	<ol style="list-style-type: none"> 1. Administrator clicks on the academic staff button from the side navigation bar. 2. Administrator clicks on the view profile button on the same row as the staff name. 3. Administrator clicks on the view report button. 4. The system shows the performance report of the selected staff.

CHAPTER 3 PROPOSED METHOD/APPROACH

Alternative flow	-
Exception flow	-

Table 3.3.2.9 View Subordinate Use Case Description

Use Case ID	00009
Use Case Name	View Subordinate
Brief Description	Superior (staff with privileges) can view other staff's records.
Actor	Superior
Trigger	Click the view all staff button from the side navigation bar.
Precondition	Superior must be logged in and have the privilege to view other staff's records, and be in the viewable staff page.
Normal flow of events	<ol style="list-style-type: none"> 1. Superior clicks the view all staff button from the side navigation bar. 2. The system displays the list of viewable staff.
Alternative flow	<ol style="list-style-type: none"> 2a. Superior enters the staff name in the search bar. 2b. The system searches for matching staff names. 2c. The system displays matched staff names.
Exception flow	-

Table 3.3.2.10 View Subordinate Profile Use Case Description

Use Case ID	00010
Use Case Name	View Subordinate Profile
Brief Description	
Actor	Superior
Trigger	Click the view profile button on the same row as the staff name.
Precondition	Superior must be logged in, have the privilege to view other staff's records, and be in the viewable staff page.
Normal flow of events	<ol style="list-style-type: none"> 1. Superior clicks the view all staff button from the side navigation bar. 2. The system displays the list of viewable staff. 3. Superior clicks the view profile button on the same row of staff names. 4. The system displays the profile of the staff.
Alternative flow	-

CHAPTER 3 PROPOSED METHOD/APPROACH

Exception flow	-
----------------	---

Table 3.3.2.11 View Subordinate Performance Report Use Case Description

Use Case ID	00011
Use Case Name	View Subordinate Performance Report
Brief Description	Superior can view the performance report of viewable staff.
Actor	Superior
Trigger	Click the view report button from the profile of the staff.
Precondition	Superior must be logged in and have the privilege to view other staff's records, and be in the viewable staff page.
Normal flow of events	<ol style="list-style-type: none"> 1. Superior clicks the view profile button on the same row as the staff name. 2. The system displays the profile of the staff. 3. Superior clicks the view report button. 4. The system displays the staff's performance report.
Alternative flow	-
Exception flow	-

Table 3.3.2.12 View Subordinate Entries Summary Use Case Description

Use Case ID	00012
Use Case Name	View Subordinate Entries Summary
Brief Description	
Actor	Superior
Trigger	Click the view entries summary button in the view all staff page.
Precondition	Superior must be logged in and have the privilege to view other staff's records, and be in the viewable staff page.
Normal flow of events	<ol style="list-style-type: none"> 1. Superior clicks the view all staff button from the side navigation bar. 2. Superior clicks the view entries summary. 3. The system displays the staff entries summary in a table form.
Alternative flow	<ol style="list-style-type: none"> 3a. Superior clicks the sort alphabetically button. 3b. The system displays the summary of staff entries in alphabetical order. 3c. Superior clicks the sort by total entries (low to high) button.

CHAPTER 3 PROPOSED METHOD/APPROACH

	<p>3d. The system displays the summary of staff entries in ascending order based on total entries.</p> <p>3e. Superior clicks the sort by total entries (high to low) button.</p> <p>3f. The system displays the summary of staff entries in descending order based on total entries.</p>
Exception flow	-

Table 3.3.2.13 Add Faculty Use Case Description

Use Case ID	00013
Use Case Name	Add Faculty
Brief Description	Administrator can add a new faculty to the system.
Actor	Administrator
Trigger	Click the add faculty button in the faculty page.
Precondition	Administrator must be logged in and be in the faculty page.
Normal flow of events	<ol style="list-style-type: none"> 1. Administrator clicks the add faculty button. 2. Administrator enters the name of the new faculty. 3. Administrator clicks the add button to submit the form.
Alternative flow	-
Exception flow	2a. The system returns an error message if the faculty name is not filled in.

Table 3.3.2.14 Edit Faculty Use Case Description

Use Case ID	00014
Use Case Name	Edit Faculty
Brief Description	Administrator can edit the faulty name.
Actor	Administrator
Trigger	Click the edit faculty button which shows as a pen icon on the same row as faculty name.
Precondition	Administrator must be logged in and be in the faculty page.
Normal flow of events	<ol style="list-style-type: none"> 1. Administrator the edit faculty button which shows as a pen icon on the same row as the faculty name. 2. Administrator modifies or re-enters the name of the faculty. 3. Administrator clicks the save button to submit the form.
Alternative flow	-

CHAPTER 3 PROPOSED METHOD/APPROACH

Exception flow	-
----------------	---

Table 3.3.2.15 Delete Faculty Use Case Description

Use Case ID	00015
Use Case Name	Delete Faculty
Brief Description	Administrator can delete a faculty from the system.
Actor	Administrator
Trigger	Click the delete button which shows as a trash icon on the same row as the faculty name.
Precondition	Administrator must be logged in and is in the faculty page.
Normal flow of events	<ol style="list-style-type: none"> 1. Administrator clicks on the delete button which shows as a trash icon on the same row as the faculty name. 2. A confirmation modal will pop up to let the administrator confirm the deletion of the faculty. 3. Administrator clicks on the confirm delete button. 4. The faculty will be deleted.
Alternative flow	<ol style="list-style-type: none"> 3a. Administrator clicks on the cancel button. 3b. The system closes the confirmation modal. No deletion occurs and administrator is returned to the faculties page.
Exception flow	-

Table 3.3.2.16 Add Department Use Case Description

Use Case ID	00016
Use Case Name	Add Department
Brief Description	Administrator can add a new department to a specific faculty in the system.
Actor	Administrator
Trigger	Click the add department button which shows as a plus icon on the same row as the department name.
Precondition	Administrator must be logged in and is in the faculty page.
Normal flow of events	<ol style="list-style-type: none"> 1. Administrator clicks the add department button on the same row as the department name. 2. Administrator enters the name of the new department. 3. Administrator clicks the add button to submit the form.

CHAPTER 3 PROPOSED METHOD/APPROACH

Alternative flow	-
Exception flow	2a. The system returns an error if the department name is not filled in.

Table 3.3.2.17 Edit Department Use Case Description

Use Case ID	00017
Use Case Name	Edit Department
Brief Description	Administrator can edit the department name.
Actor	Administrator
Trigger	Click the edit department button which shows as a pen icon on the same row as the department name.
Precondition	Administrator must be logged in and be in the faculty page.
Normal flow of events	<ol style="list-style-type: none"> 1. Administrator clicks on the edit department button, which shows as a pen icon on the same row as the department name. 2. Administrator modifies or re-enters the name of the department. 3. Administrator clicks the save button to submit the form.
Alternative flow	-
Exception flow	-

Table 3.3.2.18 Delete Department Use Case Description

Use Case ID	00018
Use Case Name	Delete Department
Brief Description	Administrator can delete a department under a faculty from the system.
Actor	Administrator
Trigger	Click the delete button which shows as a trash icon on the same row as the department name.
Precondition	Administrator must be logged in and be in the faculty page.
Normal flow of events	<ol style="list-style-type: none"> 1. Administrator clicks the delete button which shows as a trash icon on the same row as the department name. 2. A confirmation modal will pop up to let the admin confirm the deletion of the department. 3. Administrator clicks on the confirm delete button. 4. The department will be deleted.

CHAPTER 3 PROPOSED METHOD/APPROACH

Alternative flow	3a. Administrator clicks on the cancel button. 3b. The system closes the confirmation modal. No deletion occurs and administrator is returned to the faculties page.
Exception flow	-

Table 3.3.2.19 Add Staff Use Case Description

Use Case ID	00019
Use Case Name	Add Staff
Brief Description	Administrator can add a new staff account to the system.
Actor	Administrator
Trigger	Click the add staff button in the academic staff page.
Precondition	Administrator must be logged in and is in the academic staff page.
Normal flow of events	<ol style="list-style-type: none"> 1. Administrator clicks the academic staff button from the side navigation bar. 2. Administrator clicks the add staff button in the academic staff page. 3. A form modal pop up for the administrator to fill in. 4. Administrator enters information like staff name, email, password, faculty, department, and designation, and submits the form.
Alternative flow	-
Exception flow	-

Table 3.3.2.20 Delete Staff Use Case Description

Use Case ID	00020
Use Case Name	Delete Staff
Brief Description	Administrator can delete a staff account from the system.
Actor	Administrator
Trigger	Click the delete button which shows a trash icon on the same row as the staff name.
Precondition	Administrator must be logged in and be in the academic staff page.
Normal flow of events	<ol style="list-style-type: none"> 1. Administrator clicks the delete button which shows as a trash icon on the same row as the staff name. 2. A confirmation modal will pop up to let the admin confirm the

CHAPTER 3 PROPOSED METHOD/APPROACH

	<p>deletion of the staff.</p> <p>3. Administrator clicks on the confirm delete button.</p> <p>4. The staff will be deleted.</p>
Alternative flow	<p>3a. Administrator clicks on the cancel button.</p> <p>3b. The system closes the confirmation modal. No deletion occurs and the administrator is returned to the academic staff page.</p>
Exception flow	-

Table 3.3.2.21 Assign Privilege Use Case Description

Use Case ID	00021
Use Case Name	Assign Privilege
Brief Description	Administrator can assign privileges to staff to view other staff's records.
Actor	Administrator
Trigger	Click the assign privilege button in green color on the same row of the staff name.
Precondition	Administrator must be logged in and be in the faculty page.
Normal flow of events	<p>1. Administrator clicks the view staff button on the same row as the department name.</p> <p>2. Administrator clicks the assign privilege button on the same row as the staff name.</p> <p>3. Administrator fills in the role of the staff.</p> <p>4. Administrator selects the viewable staff.</p> <p>5. Administrator clicks the assign privilege button.</p>
Alternative flow	-
Exception flow	<p>3a. If the role field is empty, the system returns an error message.</p> <p>4a. Administrator can enter a staff name in the search bar.</p> <p>4b. The system will filter and displays matching staff names.</p>

Table 3.3.2.22 Disable Privilege Use Case Description

Use Case ID	00022
Use Case Name	Disable Privilege
Brief Description	Administrator can remove the privilege of staff to view other staff's records.

Actor	Administrator
Trigger	Click the unassign privilege button in red color on the same row as the staff name.
Precondition	Administrator must be logged in and be in the faculty page.
Normal flow of events	1. Administrator clicks the unassign privilege button on the same row of a specific staff name.
Alternative flow	-
Exception flow	-

Table 3.3.2.23 Edit Performance Area Weightage Use Case Description

Use Case ID	00023
Use Case Name	Edit Performance Area Weightage
Brief Description	Administrator can modify the weightage of each performance area.
Actor	Administrator
Trigger	Change the value for a specific performance area.
Precondition	Administrator must be logged in and in the performance area page.
Normal flow of events	1. Administrator clicks the performance area button from the side navigation bar. 2. Administrator re-enters a value for a specific performance area.
Alternative flow	-
Exception flow	-

3.3.3 Program Development

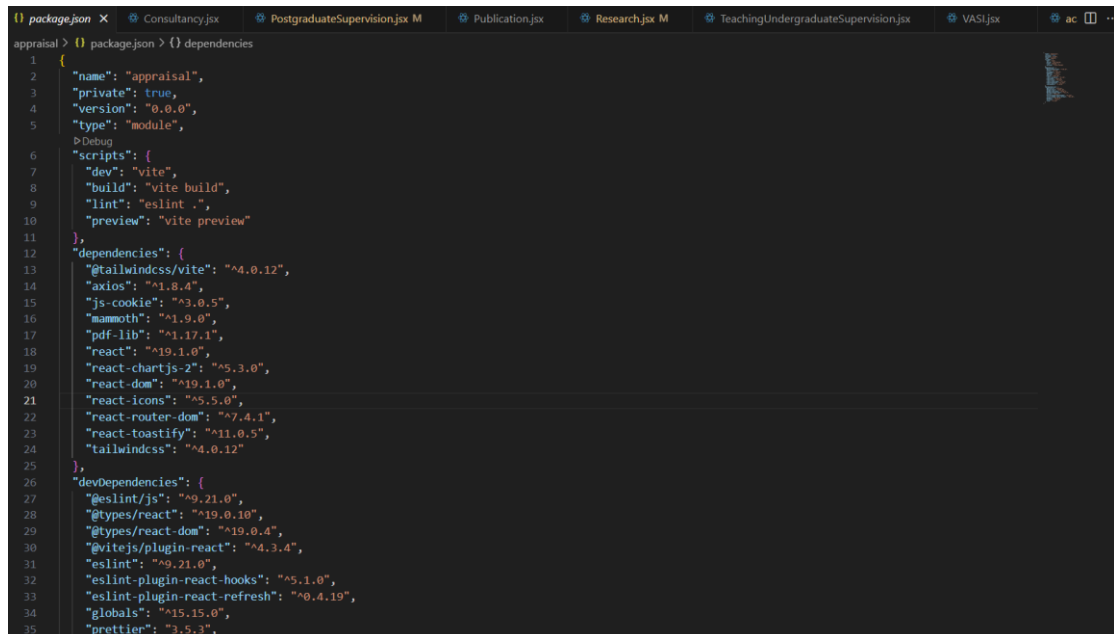
3.3.3.1 Server-side Development

The MERN stack architecture is used in the development of this project, which incorporates both client-side and server-side technologies. Figure 3.3.3.1.1 and Figure 3.3.3.1.2 show all the dependencies used in the frontend and backend for this project. For the client-side, React.js is used to create a responsive and dynamic user interface, while Tailwind CSS is used for styling purposes. Axios is used to send asynchronous HTTP requests to the backend, while the react-router-dom handles routing between pages, like login and profile. On the other hand, user notifications are set up using the react-toastify which provides feedback of user action. The system depends on js-cookie to manage sessions and the JWT for handling authentication. The token

generated will be stored in HTTP-only cookies.

For the server-side, Node.js is the runtime environment, while Express.js provides a framework for managing middleware and route handling. Express routers are used to map the API endpoint received from the frontend to the respective controller functions. All the routes.js files used in this project are categorized under the routes folder as shown in Figure 3.3.3.1.3. All the data is persistently stored in the MongoDB database. Mongoose is used to model the application data stored in collections for faculties, departments, staff, and their privileges, defining the schemas and the methods to use for database operations like findById and findByIdAndDelete. Environment variables like database connection strings and JWT secrets are encrypted and stored in a .env file and loaded via the dotenv package.

The service.js, as shown in Figure 3.3.3.1.5, is the main code file to control the backend. It involves server initialization, database connection, and registration of middleware and routes. These routes handle GET, POST, PUT, and DELETE requests sent from the frontend, perform logic such as validating data, querying MongoDB, and returning JSON responses or status messages accordingly.



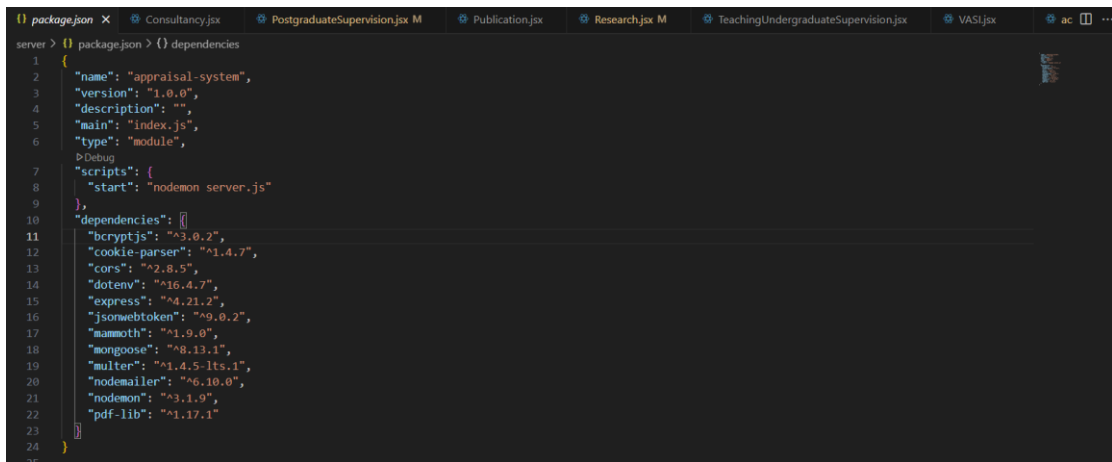
```

1 {
2   "name": "appraisal",
3   "private": true,
4   "version": "0.0.0",
5   "type": "module",
6   "scripts": {
7     "dev": "vite",
8     "build": "vite build",
9     "lint": "eslint .",
10    "preview": "vite preview"
11  },
12  "dependencies": {
13    "@tailwindcss/vite": "^4.0.12",
14    "axios": "^1.8.4",
15    "js-cookie": "^3.0.5",
16    "mammoth": "^1.9.0",
17    "pdf-lib": "^1.17.1",
18    "react": "^19.1.0",
19    "react-chartjs-2": "^5.3.0",
20    "react-dom": "^19.1.0",
21    "react-icons": "^5.5.0",
22    "react-router-dom": "^7.4.1",
23    "react-toastify": "^11.0.5",
24    "tailwindcss": "^4.0.12"
25  },
26  "devDependencies": {
27    "@eslint/js": "^9.21.0",
28    "@types/react": "^19.0.10",
29    "@types/react-dom": "^19.0.4",
30    "@vitejs/plugin-react": "^4.3.4",
31    "eslint": "^9.21.0",
32    "eslint-plugin-react-hooks": "^5.1.0",
33    "eslint-plugin-react-refresh": "^0.4.19",
34    "globals": "^15.15.0",
35    "prettier": "3.5.3",

```

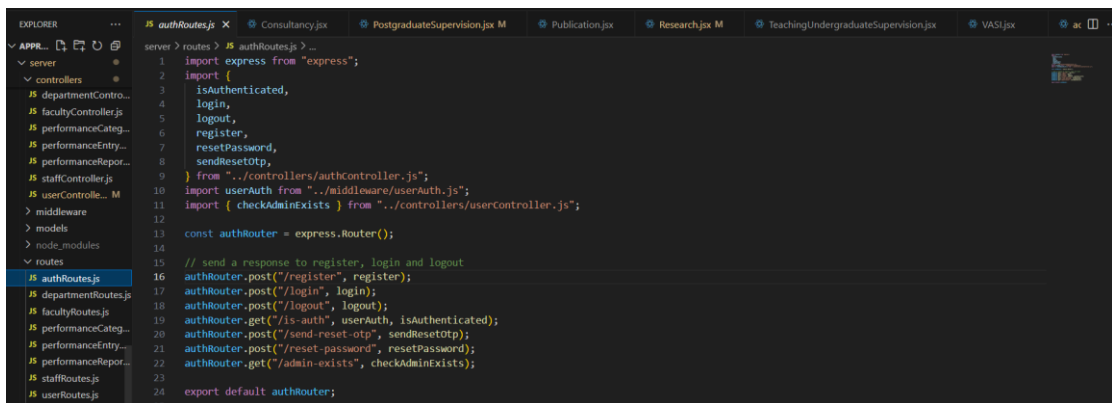
Figure 3.3.3.1.1 package.json (Frontend)

CHAPTER 3 PROPOSED METHOD/APPROACH



```
1 {
2   "name": "appraisal-system",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   "type": "module",
7   "scripts": {
8     "start": "nodemon server.js"
9   },
10  "dependencies": {
11    "bcryptjs": "^3.0.2",
12    "cookie-parser": "^1.4.7",
13    "cors": "^2.8.5",
14    "dotenv": "^16.4.7",
15    "express": "^4.21.2",
16    "jsonwebtoken": "^9.0.2",
17    "mammoth": "^1.9.0",
18    "mongoose": "^8.13.1",
19    "multer": "^1.4.5-lts.1",
20    "nodemailer": "^6.10.0",
21    "nodemon": "^3.1.9",
22    "pdf-lib": "^1.17.1"
23  }
24 }
```

Figure 3.3.3.1.2 package.json (Backend)



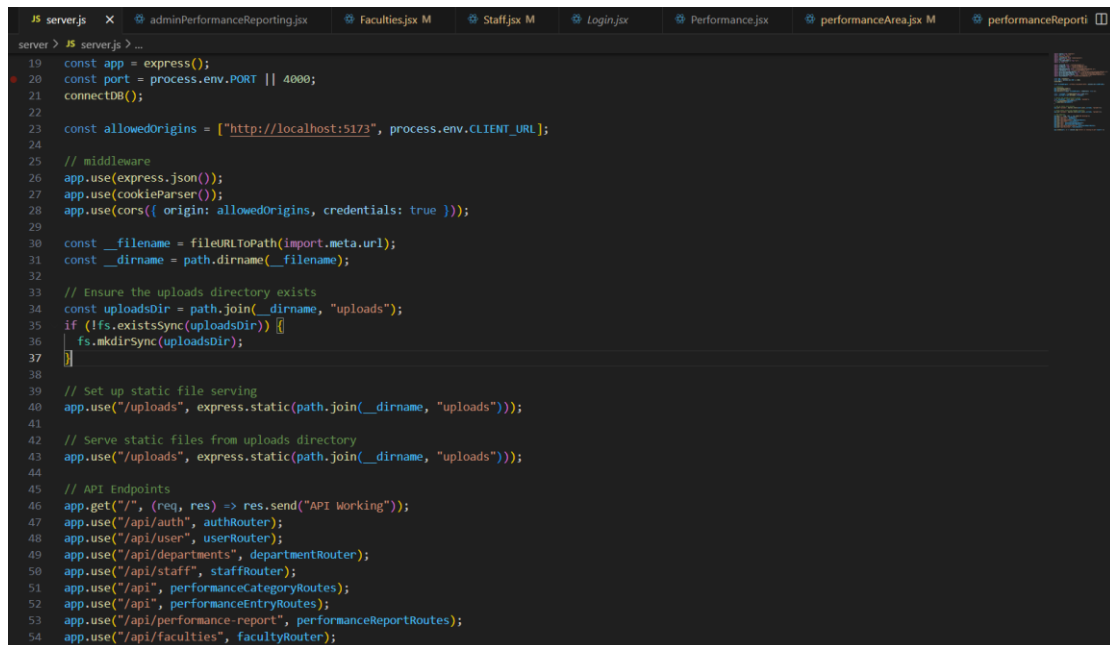
```
1 import express from "express";
2 import {
3   isAuthenticated,
4   login,
5   logout,
6   register,
7   resetPassword,
8   sendResetotp,
9 } from "../controllers/authController.js";
10 import userAuth from "../middleware/userAuth.js";
11 import { checkAdminExists } from "../controllers/userController.js";
12
13 const authRouter = express.Router();
14
15 // send a response to register, login and logout
16 authRouter.post("/register", register);
17 authRouter.post("/login", login);
18 authRouter.post("/logout", logout);
19 authRouter.get("/is-auth", userAuth, isAuthenticated);
20 authRouter.post("/send-reset-otp", sendResetotp);
21 authRouter.post("/reset-password", resetPassword);
22 authRouter.get("/admin-exists", checkAdminExists);
23
24 export default authRouter;
```

Figure 3.3.3.1.3 routes Folder and authRoute.js



```
1 import mongoose from "mongoose";
2
3 Windsurf: Refactor | Explain | Generate JSDoc | X
4 const connectDB = async () => {
5   mongoose.connection.on("connected", () => console.log("Database Connected"));
6
7   await mongoose.connect(process.env.MONGODB_URI, {
8     useNewUrlParser: true,
9     useUnifiedTopology: true,
10   }); // Connect to MongoDB using the URI from environment variables
11 }
12
13 export default connectDB;
```

Figure 3.3.3.1.4 mongodb.js



```

server > JS server.js > ...
19 const app = express();
20 const port = process.env.PORT || 4000;
21 connectDB();
22
23 const allowedOrigins = ["http://localhost:5173", process.env.CLIENT_URL];
24
25 // middleware
26 app.use(express.json());
27 app.use(cookieParser());
28 app.use(cors({ origin: allowedOrigins, credentials: true }));
29
30 const __filename = fileURLToPath(import.meta.url);
31 const __dirname = path.dirname(__filename);
32
33 // Ensure the uploads directory exists
34 const uploadsDir = path.join(__dirname, "uploads");
35 if (!fs.existsSync(uploadsDir)) {
36   fs.mkdirSync(uploadsDir);
37 }
38
39 // Set up static file serving
40 app.use("/uploads", express.static(path.join(__dirname, "uploads")));
41
42 // Serve static files from uploads directory
43 app.use("/uploads", express.static(path.join(__dirname, "uploads")));
44
45 // API Endpoints
46 app.get("/", (req, res) => res.send("API Working"));
47 app.use("/api/auth", authRouter);
48 app.use("/api/user", userRouter);
49 app.use("/api/departments", departmentRouter);
50 app.use("/api/staff", staffRouter);
51 app.use("/api", performanceCategoryRoutes);
52 app.use("/api", performanceEntryRoutes);
53 app.use("/api/performance-report", performanceReportRoutes);
54 app.use("/api/faculties", facultyRouter);

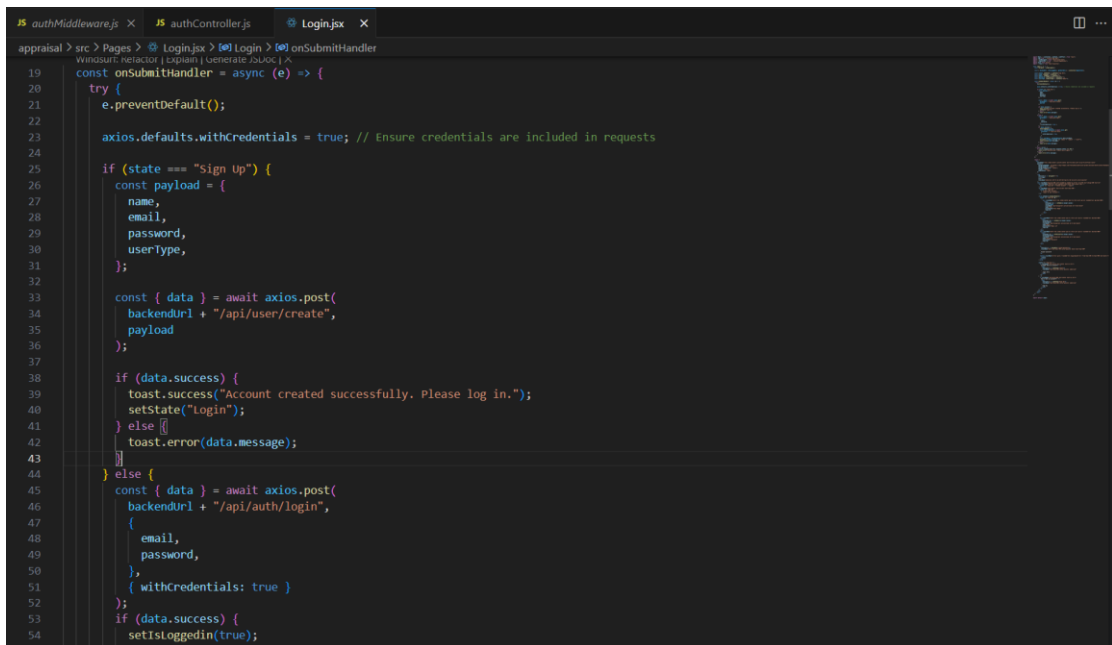
```

Figure 3.3.3.1.5 server.js

3.3.3.2 Login Function Development

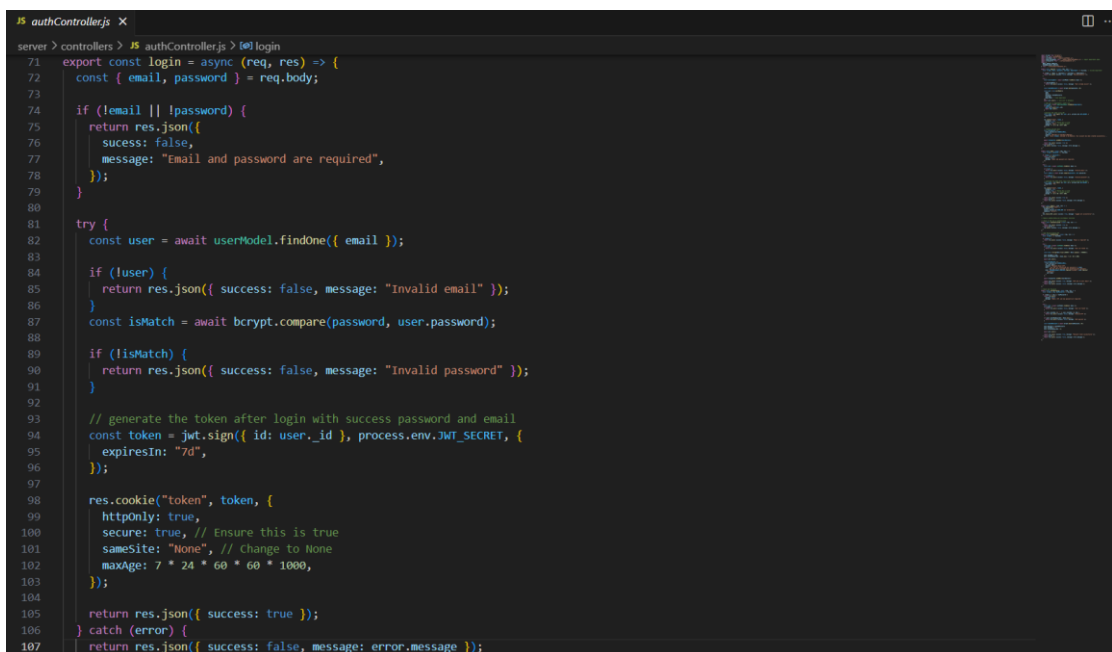
When users enter their email and password on the login page, the inputs are managed using `useState` and an HTTP POST request is sent to the backend. The backend will check the credentials by validating the email and comparing the password using `bcrypt`. A JWT token is created upon successful validation of credentials using the user's ID and a secret key. This token is then saved in an HTTP-only cookie with secure configuration (`httpOnly: true`, `secure: true`, `sameSite: "None"`, and `maxAge` for 7 days). This can prevent access to tokens from the client-side JavaScript and offers better protection from XSS attacks. Upon successful login of users, the login state will be set to `true`, and a follow-up request will be made to fetch the user's information. Users will be redirected to the appropriate dashboard depending on the user's type. To protect routes by ensuring only logged in users can access, a middleware is used in the backend. This middleware checks for the presence of a valid token in the cookie. It verifies the token and retrieves the user's information from the database and attaches it to `req.user`.

CHAPTER 3 PROPOSED METHOD/APPROACH



```
19  const onSubmitHandler = async (e) => {
20    try {
21      e.preventDefault();
22
23      axios.defaults.withCredentials = true; // Ensure credentials are included in requests
24
25      if (state === "Sign Up") {
26        const payload = {
27          name,
28          email,
29          password,
30          userType,
31        };
32
33        const { data } = await axios.post(
34          backendUrl + "/api/user/create",
35          payload
36        );
37
38        if (data.success) {
39          toast.success("Account created successfully. Please log in.");
40          setState("Login");
41        } else {
42          toast.error(data.message);
43        }
44      } else {
45        const { data } = await axios.post(
46          backendUrl + "/api/auth/login",
47          {
48            email,
49            password,
50          },
51          { withCredentials: true }
52        );
53        if (data.success) {
54          setIsLoggedIn(true);
55        }
56      }
57    } catch (error) {
58      toast.error(error.message);
59    }
60  }
61}
```

Figure 3.3.3.2.1 Login.jsx (Handling login/signup submission)



```
71 export const login = async (req, res) => {
72   const { email, password } = req.body;
73
74   if (!email || !password) {
75     return res.json({
76       success: false,
77       message: "Email and password are required",
78     });
79   }
80
81   try {
82     const user = await userModel.findOne({ email });
83
84     if (!user) {
85       return res.json({ success: false, message: "Invalid email" });
86     }
87     const isMatch = await bcrypt.compare(password, user.password);
88
89     if (!isMatch) {
90       return res.json({ success: false, message: "Invalid password" });
91     }
92
93     // generate the token after login with success password and email
94     const token = jwt.sign({ id: user._id }, process.env.JWT_SECRET, {
95       expiresIn: "7d",
96     });
97
98     res.cookie("token", token, {
99       httpOnly: true,
100       secure: true, // Ensure this is true
101       sameSite: "None", // Change to None
102       maxAge: 7 * 24 * 60 * 60 * 1000,
103     });
104
105     return res.json({ success: true });
106   } catch (error) {
107     return res.json({ success: false, message: error.message });
108   }
109 }
```

Figure 3.3.3.2.2 authController.js (Login function)

```

1  import jwt from "jsonwebtoken";
2  import userModel from "../models/userModel.js";
3
4  export const isAuthenticated = async (req, res, next) => {
5    try {
6      const token = req.cookies.token;
7      if (!token) {
8        return res.json({ success: false, message: "Not authenticated" });
9      }
10
11      const decoded = jwt.verify(token, process.env.JWT_SECRET);
12      req.user = await userModel.findById(decoded.id);
13      next();
14    } catch (error) {
15      res.json({ success: false, message: error.message });
16    }
17  };

```

Figure 3.3.3.2.3 authMiddleware.js

3.3.3.3 Faculty and Department Management Function Development

The list of faculties and departments is displayed on the faculty page for the administrator. Form modals are provided to perform create, edit, and delete operations. When the administrator submits a form to add a faculty or department, the HTTP POST request is triggered via axios to the corresponding backend API endpoint. This request sends the form data and uses `withCredentials: true` to include the JWT token stored in the browser cookies for authentication purposes. Express on the backend will handle the request. A JSON response will be sent back to the client to indicate the success or failure of the action performed. If the administrator chooses to edit the current faculty or department names, the ID of the faculty or department and the new name will be passed to the backend using HTTP PUT. For the delete operation, a confirmation modal will first pop up. If the administrator confirms the operation, the faculty or department ID in the URL path and JWT token will be passed through an HTTP DELETE request to the backend. In MongoDB, the document is located by `_id` and it will be permanently deleted from the faculty or department collection.

```

177 const addDepartment = async (facultyId, departmentName) => {
178   try {
179     const { data } = await axios.post(
180       `http://localhost:4000/api/faculties/${facultyId}/departments/add`,
181       { name: departmentName },
182       { withCredentials: true }
183     );
184
185     if (data.success) {
186       setFaculties((prevFaculties) =>
187         prevFaculties.map((faculty) =>
188           faculty_id === facultyId ? data.faculty : faculty
189         )
190       );
191       toast.success("Department added successfully.");
192       setShowAddDepartmentModal(false);
193       setNewDepartmentName("");
194     } else {
195       toast.error(data.message);
196     }
197   } catch (error) {
198     toast.error("Failed to add department.");
199   }
200 }
201
202 const editFaculty = async (facultyId, facultyName) => {
203   try {
204     const { data } = await axios.put(
205       `http://localhost:4000/api/faculties/${facultyId}`,
206       { name: facultyName },
207       { withCredentials: true }
208     );
209
210     if (data.success) {
211       setFaculties((prevFaculties) =>
212         prevFaculties.map((faculty) =>

```

Figure 3.3.3.3.1 Faculties.jsx (with function to handle operation for department)

```

4 export const addFaculty = async (req, res) => {
5   try {
6     const { name } = req.body;
7     const faculty = new facultyModel({ name, departments: [] }); // Initialize departments as an empty array
8     await faculty.save();
9     res.json({ success: true, faculty });
10  } catch (error) {
11    res.json({ success: false, message: error.message });
12  }
13 }
14
15 export const removeFaculty = async (req, res) => {
16   try {
17     const { id } = req.params;
18     await facultyModel.findByIdAndDelete(id);
19     res.json({ success: true, message: "Faculty removed" });
20  } catch (error) {
21    res.json({ success: false, message: error.message });
22  }
23 }
24
25 export const getFaculties = async (req, res) => {
26   try {
27     const faculties = await facultyModel.find().populate({
28       path: "departments",
29       populate: { path: "staff" },
30     }); // populate departments field
31     res.json({ success: true, faculties });
32  } catch (error) {
33    res.json({ success: false, message: error.message });
34  }
35 }
36

```

Figure 3.3.3.3.2 facultyController.js

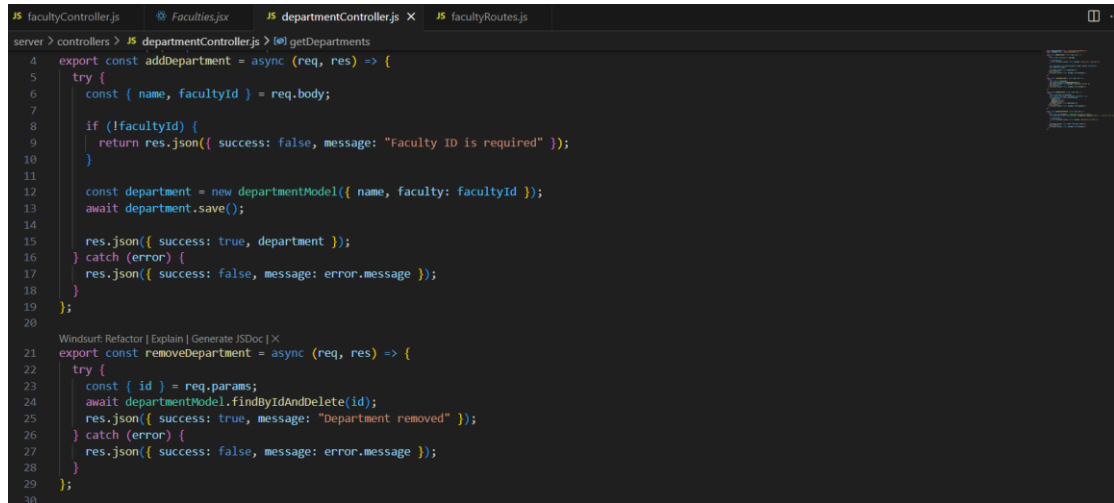


Figure 3.3.3.3.3 departmentController.js

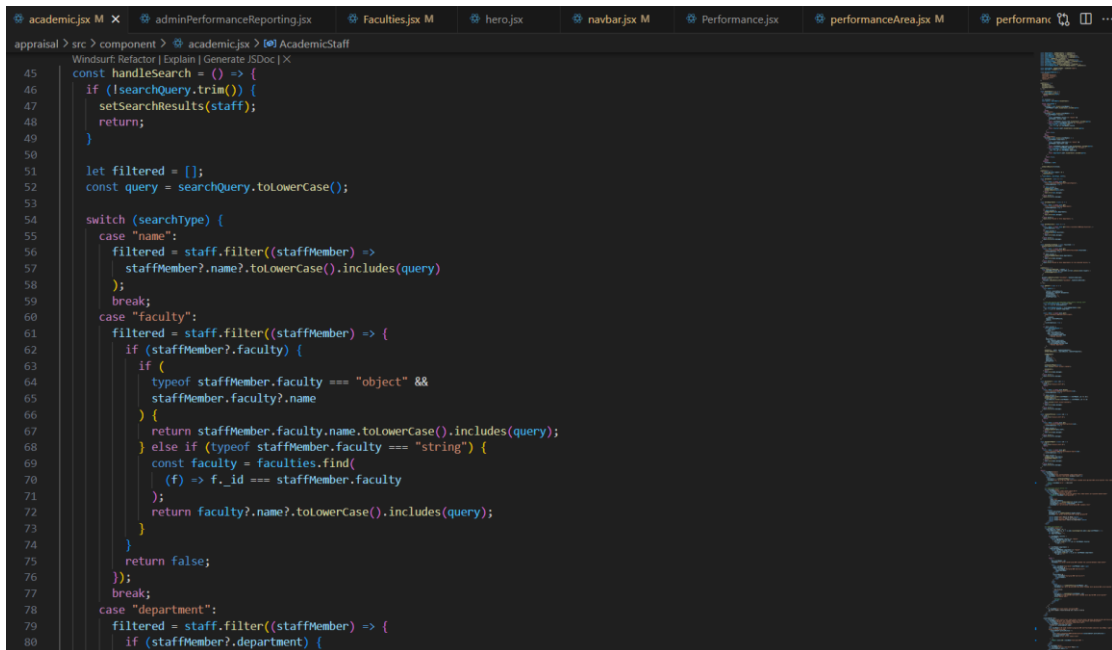
3.3.3.4 Staff Management Function Development

For the staff management function, the administrator can add, view, search, and delete staff. After submitting the add staff form, an HTTP POST request is sent via axios to the respective API endpoint. The backend controller will first hash the password using bcrypt.js, create a staff document, and update the department by adding to the staff array. If the administrator chooses to delete a staff, an HTTP DELETE request along with the URL containing the staff ID is sent upon clicking the delete button. After verifying the ID, the staff will be removed from the database. Besides, the administrator can also search staff by name, department, or faculty. A search is carried out by initiating an HTTP GET request containing the query string to the specific backend API endpoint. The backend then processes the request. The function in the controller prepares a MongoDB query that makes use of the \$regex operator to do a case-insensitive pattern match on the staff name field.

Another feature included under the staff management function is assigning or removing privileges for selected staff members. The administrator can assign privileges to selected staff (superior) if there are staff under them. In the faculty page, the administrator can view the staff under each department. On the same row as each staff name, there is a button to assign privileges. The button is green in color if no privilege is assigned to the staff and red in color if the privilege is assigned. To remove the privilege, the administrator can click on the red button. Upon the submission of

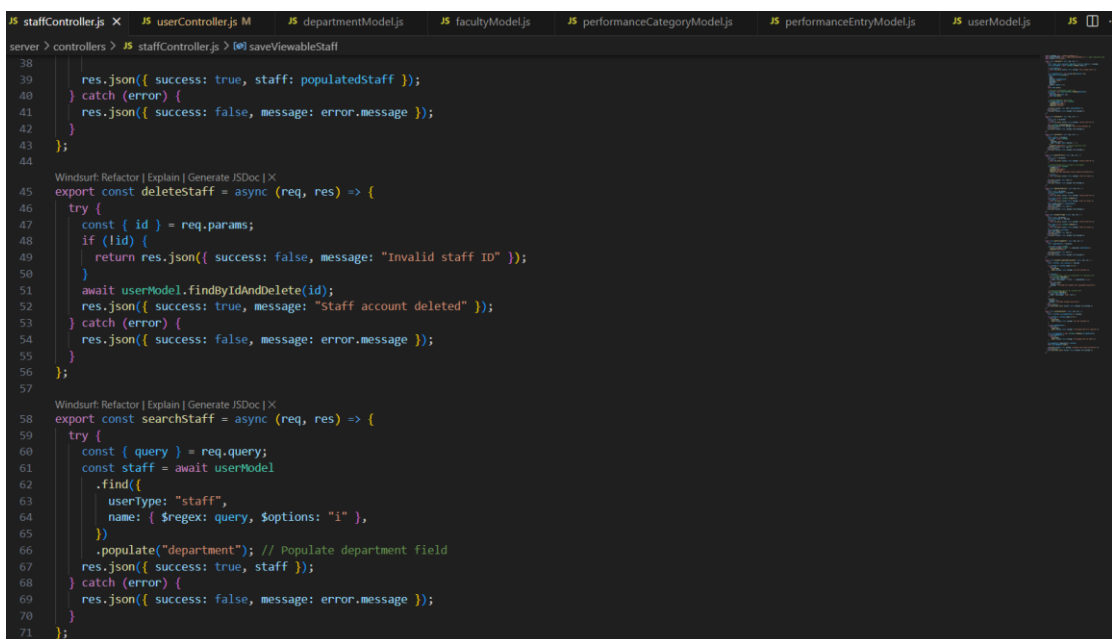
CHAPTER 3 PROPOSED METHOD/APPROACH

assigning or removing privilege, an HTTP PUT request is sent to the backend. For assigning privilege, the request will include the role of the staff and the list of staff IDs. The backend then processes the request and performs an update in MongoDB.



```
45 const handleSearch = () => {
46   if (!searchQuery.trim()) {
47     setSearchResults(staff);
48     return;
49   }
50
51   let filtered = [];
52   const query = searchQuery.toLowerCase();
53
54   switch (searchType) {
55     case "name":
56       filtered = staff.filter((staffMember) =>
57         staffMember?.name?.toLowerCase().includes(query)
58       );
59       break;
60     case "faculty":
61       filtered = staff.filter((staffMember) => {
62         if (staffMember?.faculty) {
63           if (
64             typeof staffMember.faculty === "object" &&
65             staffMember.faculty?.name
66           ) {
67             return staffMember.faculty.name.toLowerCase().includes(query);
68           } else if (typeof staffMember.faculty === "string") {
69             const faculty = faculties.find(
70               (f) => f._id === staffMember.faculty
71             );
72             return faculty?.name?.toLowerCase().includes(query);
73           }
74         }
75         return false;
76       });
77       break;
78     case "department":
79       filtered = staff.filter((staffMember) => {
80         if (staffMember?.department) {
```

Figure 3.3.3.4.1 academic.jsx (Search staff)



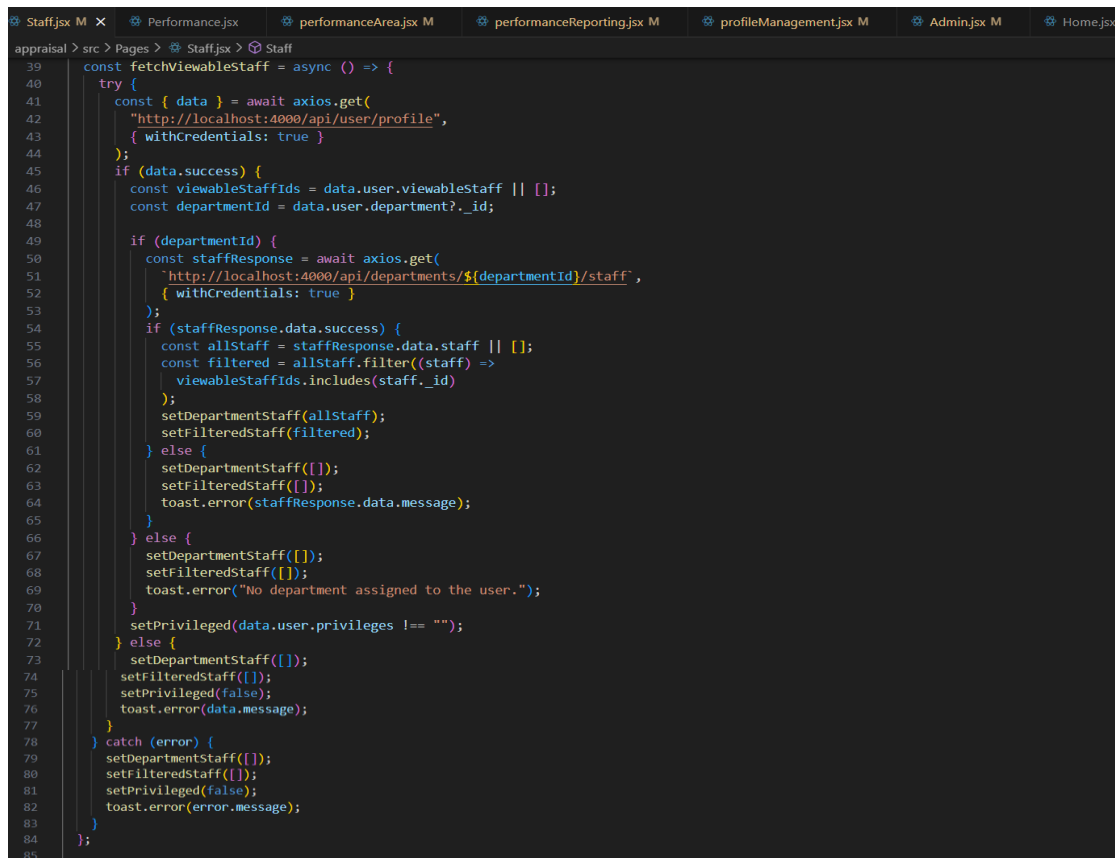
```
38 res.json({ success: true, staff: populatedStaff });
39 } catch (error) {
40   res.json({ success: false, message: error.message });
41 }
42 };
43
44
45 Windsurf: Refactor | Explain | Generate JSDoc | X
46 export const deleteStaff = async (req, res) => {
47   try {
48     const { id } = req.params;
49     if (!id) {
50       return res.json({ success: false, message: "Invalid staff ID" });
51     }
52     await userModel.findByIdAndDelete(id);
53     res.json({ success: true, message: "Staff account deleted" });
54   } catch (error) {
55     res.json({ success: false, message: error.message });
56   }
57 };
58
59 Windsurf: Refactor | Explain | Generate JSDoc | X
60 export const searchStaff = async (req, res) => {
61   try {
62     const { query } = req.query;
63     const staff = await userModel
64       .find({
65         userType: "staff",
66         name: { $regex: query, $options: "i" },
67       })
68       .populate("department"); // Populate department field
69     res.json({ success: true, staff });
70   } catch (error) {
71     res.json({ success: false, message: error.message });
72   }
73 };
```

Figure 3.3.3.4.2 staffController.js (Delete and Search controller function)

3.3.3.5 Viewable Staff Function Development

The 'Viewable Staff' function allows superiors (selected privileged staff) to view the profiles and the performance-related information of selected staff. When a staff login to the system, an HTTP GET request is sent to retrieve the staff's data. Critical fields of this response include privileges, which indicate whether the staff have access to view other staff's records, and the department of the staff. Once the staff's department ID is retrieved, another HTTP GET request will be sent, and the response will be all staff who belong to that specific department. Then, this staff list will be filtered to ensure that only the staff whose ID matches is left.

In the side navigation bar of the staff interface, there will be a view all staff button if the staff is a superior. The superiors can view staff who are under them and view each staff profiles and performance summaries.



```

39  const fetchViewableStaff = async () => {
40    try {
41      const { data } = await axios.get(
42        "http://localhost:4000/api/user/profile",
43        { withCredentials: true }
44      );
45      if (data.success) {
46        const viewableStaffIds = data.user.viewableStaff || [];
47        const departmentId = data.user.department?._id;
48
49        if (departmentId) {
50          const staffResponse = await axios.get(
51            "http://localhost:4000/api/departments/${departmentId}/staff",
52            { withCredentials: true }
53          );
54          if (staffResponse.data.success) {
55            const allStaff = staffResponse.data.staff || [];
56            const filtered = allStaff.filter((staff) =>
57              viewableStaffIds.includes(staff._id)
58            );
59            setDepartmentStaff(allStaff);
60            setFilteredStaff(filtered);
61          } else {
62            setDepartmentStaff([]);
63            setFilteredStaff([]);
64            toast.error(staffResponse.data.message);
65          }
66        } else {
67          setDepartmentStaff([]);
68          setFilteredStaff([]);
69          toast.error("No department assigned to the user.");
70        }
71        setPrivileged(data.user.privileges !== "");
72      } else {
73        setDepartmentStaff([]);
74        setFilteredStaff([]);
75        setPrivileged(false);
76        toast.error(data.message);
77      }
78    } catch (error) {
79      setDepartmentStaff([]);
80      setFilteredStaff([]);
81      setPrivileged(false);
82      toast.error(error.message);
83    }
84  };
85

```

Figure 3.3.3.5.1 Staff.jsx (fetchViewableStaff function)

3.4 Implementation of Issues and Challenges

Some issues and challenges faced while developing the Comprehensive Employee Appraisal System.

- **Designing a Scalable Privilege System for Staff Supervision**

The requirement for a scalable privileges system, so that selected staff (e.g., Heads of Department and senior academic staff) can supervise and view the performance records of other staff in their department, was one major challenge faced during the system development. It is simple to assign privileges to a single user, however, the complexity arises for scalability and flexibility, including different levels of access, dynamically grouping viewable staff, or multiple supervisors per department for the same collection. Thus, a data model was required to store not just the privilege flag, but the privileged staff role and the list of users they can access. Therefore, some time has been spent on research in this area to ensure flexibility for the system to adapt to changes in the supervision structure in the future.

- **Different performance weightage based on staff designations and faculty**

While developing the weightage modification for each performance area, an issue faced is the implementation of different performance weightages based on the staff designation and faculty. This is due to the reason that different roles such as Professor, Associate Professor, and Lecturer, are evaluated with different emphasis across performance areas. Furthermore, different faculties may also apply different evaluation standards depending on their academic focus. Thus, more research needs to be carried out during the Final Year Project 2 to discover how to allow for future scalability in the event of changes to these weightages.

3.5 Timeline

3.5.1 Overview

The development timeline for the project was structured based on the RAD methodology, emphasizing the short planning phases, quick iterations, and constant feedback. Therefore, the project was broken into three main phases of Requirement Planning, Design & Construction, and Finalization. The objective of Final Year Project

CHAPTER 3 PROPOSED METHOD/APPROACH

1 was to finish the foundational modules of the system, such as the login module, management of staff and department, and a simple role-based access control, and set the stage for advanced features in Final Year Project 2.

The first three weeks of work focused on requirements planning, system analysis, and finalizing the tools and technology stack to be used. During this phase, the problem statements and core objectives of the project were identified. A review of the existing website and similar projects was also done to identify the strengths and weaknesses. Besides, a plan of the project timeline and tools, and the structure of the system was defined through diagramming and preliminary design of the database.

The next phase is the user design, construction, and feedback phase. In this phase, backend design and development using Node.js and MongoDB ran in parallel with frontend design and implementation using React.js and Tailwind CSS. Major modules developed within the scheduling period included a login and authentication module using JWT and cookies, staff, department, and faculty management, the privilege assignment module for role-based access, and the report generation module. Besides, the preparation of the Final Year Project report and presentation is also conducted in this phase.

Lastly, the calendar module and the system evaluation module will be developed in Final Year Project 2. All modules will be tested to ensure that the project is working as expected and to prevent errors and bugs in the system during the final product and implementation phase. In addition, a report and presentation will be prepared and presented in Final Year Project 2.

CHAPTER 3 PROPOSED METHOD/APPROACH

3.5.2 Gantt Chart

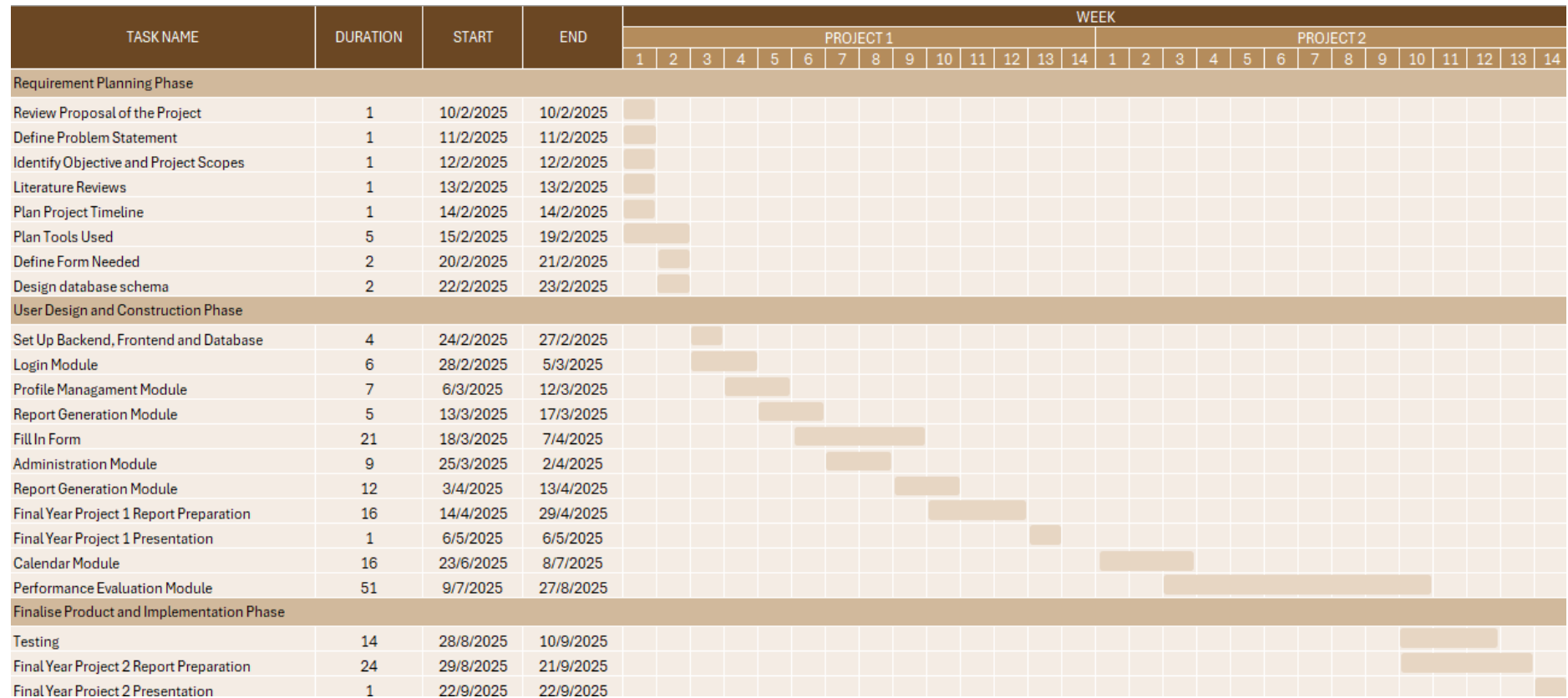


Figure 3.5.2.1 Gantt Chart of the Project Timeline

3.6 Summary

In this chapter, the RAD methodology was explained as the chosen development approach due to its ability to carry out continuous testing in each phase, reducing risk by discovering potential errors or bugs early. The ability of RAD methodology to incorporate changes is also a reason that allows for easier modification or enhancement of features. The hardware and software tools required for the development of the frontend and backend of the system are also listed in this chapter. Besides, a use case diagram is provided to illustrate the roles and functions accessible to the administrator, the staff, and the superior (privileged staff). The system development process is also explained. Lastly, a Gantt Chart is provided to show information about the timeline for developing the project and to ensure that delays in development are minimized.

CHAPTER 4 PRELIMINARY WORK

4.1 Overview

In the Final Year Project 1, some features in the modules are developed. Modules include the login module, profile management module, report generation module, and administration module. The system uses the MERN stack with layout designs by React.js and Tailwind CSS and is responsive across desktop and mobile views. There is a navbar on each page. The profile picture and the name of the user will be shown if the user is logged in. Besides, a side navigation bar is provided to direct users to respective pages upon selection of the user. Some pages are access-restricted based on the user's role. For instance, only the administrator can manage staff and assign privileges. Privileged staff can view some of the staff under their supervision, depending on the viewable staff assignment performed by the admin.

4.2 Login Page

When a user accesses the default URL of the website, the user will be redirected to the website's main page. Users can click on the login button. Users are required to log in to their accounts by entering their email and password. If the user's email is not found or the password does not match the email, the system will prompt an error message to acknowledge the user. If both the email and password are valid, the user will be issued a JWT stored in an HTTP-only cookie with an expiration of 7 days. If the user does not remember the password, they can click on the forget password. The user will need to submit their email address, and the system will send an email with a 6-digit OTP code to that email address which lasts for 15 minutes. Once the user receives the OTP code, they will need to enter the code and the new password. Upon submission, the system will first validate whether the OTP is valid. If valid, the system will hash the new password and update the database.

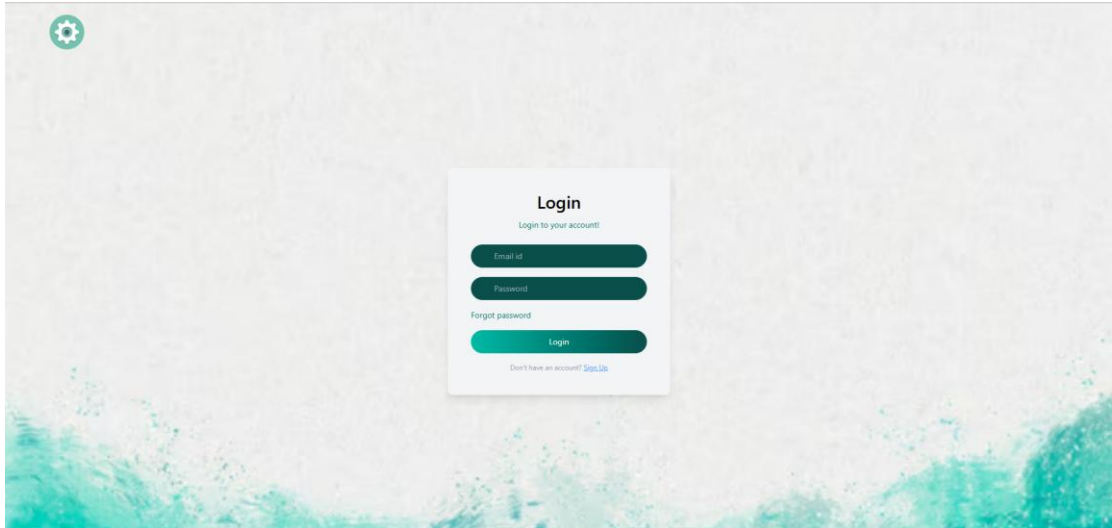


Figure 4.2.1 Login Page

4.3 User Profile

4.3.1 Staff and Superior Profile

In the staff and superior (staff with privilege) profile page, they can view their personal and job-related information such as profile picture, name, faculty, department, designation, and email. Staff and superior are not allowed to modify specific fields like email, faculty, department, and designation. Other fields, such as profile picture, name, contact number, qualifications, and area of expertise, are editable. They can edit the current information and click on the update profile button to save the latest information. The success or failure of the update will be acknowledged with a toast message displayed. In addition, users can change their password on the same page. The current password is required for validation purposes.

Appraisal

Lee Weng Hin

Staff Panel

Profile Management

Performance Reporting

Performance Area

Profile Management

Click to change photo

Name

Lee Weng Hin

Email

whlee@gmail.com

Faculty/Institute/Centre/Division

Faculty of Information and Communication Technology (FICT)

Designation

Lecturer

Contact Number

Qualifications

Bachelor

Area of Expertise (Optional)

Update Profile

Delete Profile

Figure 4.3.1.1 Profile Management Page

Change Password

Current Password

New Password

Confirm New Password

Change Password

Figure 4.3.1.2 Change Password in Profile Management Page

4.3.2 Administrator Profile

In the administrator profile page, the administrator can upload a profile picture and modify the name. The change password section is optional if the administrator would like only to update the profile.

The screenshot shows a web application interface for 'Appraisal'. At the top left is a gear icon and the word 'Appraisal'. At the top right is a user profile icon and the name 'Jean Kwan'. The main content area is titled 'Update Profile'. It features a square placeholder for a profile picture with a camera icon and the text 'Click to change photo'. Below this is a 'Name' field containing 'Jean Kwan'. Underneath is a section titled 'Change Password (Optional)' which includes three password fields: 'Current Password' (with placeholder 'Enter current password'), 'New Password' (with placeholder 'Enter new password'), and 'Confirm Password' (with placeholder 'Confirm new password'). At the bottom right of the form are two buttons: 'Cancel' and 'Update Profile'.

Figure 4.3.2.1 Administrator Profile Page

4.4 Manage Faculty and Department

In the faculties page, administrator can view all the existing faculties, departments under each faculty, and staff under a specific department. Administrator can add new faculty or a new department by clicking on the add faculty button or the plus icon for adding a department. A form will pop up which requires administrator to key in the faculty or department name. Besides, administrator can perform deletion for both faculties and departments. By clicking on the trash icon, a delete confirmation will be displayed, so administrator can choose to proceed the deletion or cancel it. Furthermore, administrator can assign privileges to selected staff by clicking the green icon on the same row as the staff name. Administrator will need to assign a role to the staff and select viewable staff, which means all the records of selected staff under the select viewable staff section will be able to be viewed by the staff with the assigned privilege. Administrator can also remove the privilege.

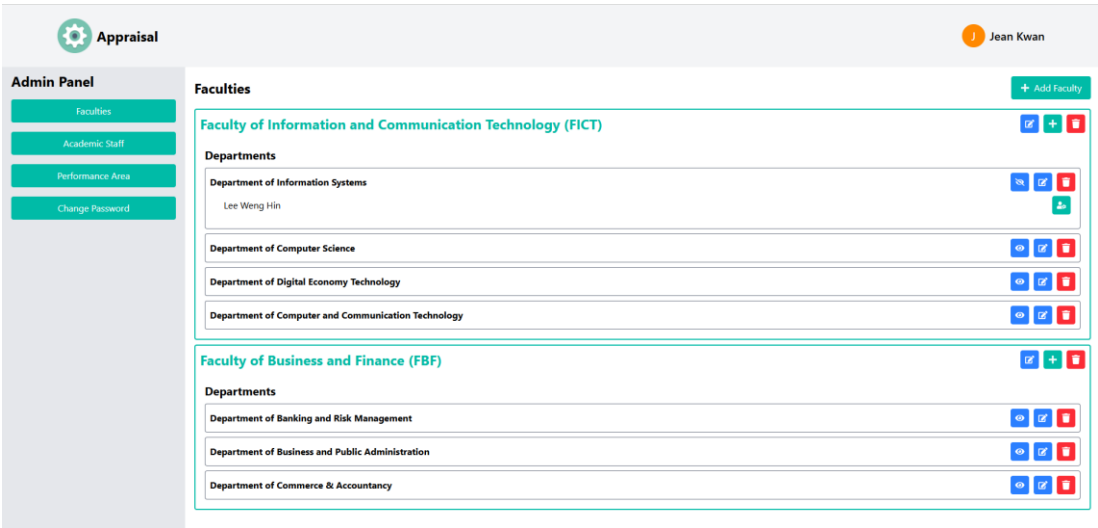


Figure 4.4.1 Faculties Page

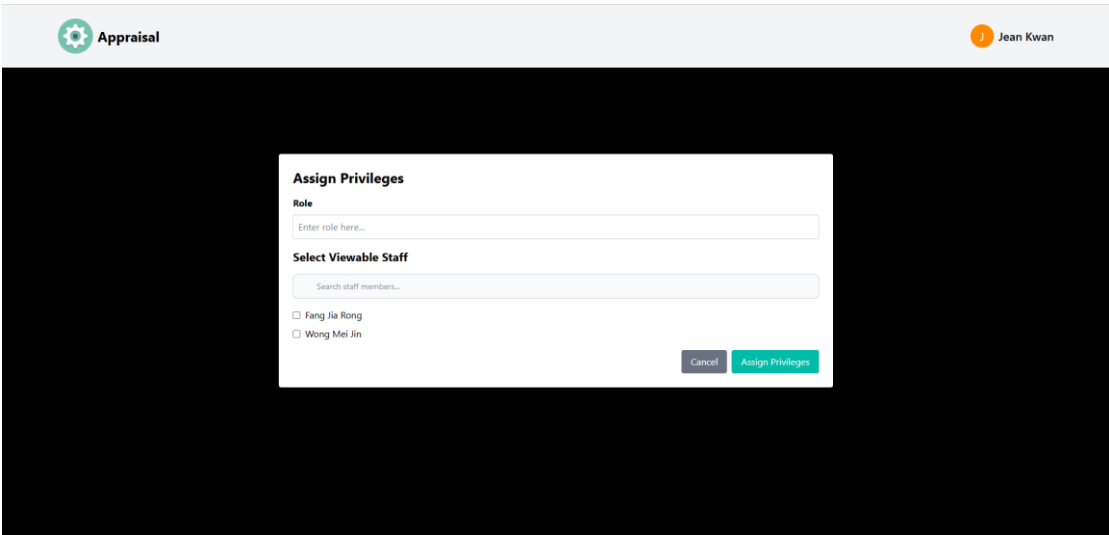


Figure 4.4.2 Assign Privilege Form

4.5 Manage Staff

To avoid any new creation of accounts by others, only administrator can add a new account, which is by adding a new staff. Some important information, such as name, email address, password, faculty, and designation, is needed to add a new staff. A search feature is implemented so that administrator can search staff more easily. Administrator can search staff by switching between three categories, which are search by name, search by department, and search by faculty. Moreover, administrator can

CHAPTER 4 PRELIMINARY WORK

view the detailed information of staff by clicking on the view profile button. This will display the profile of staff with personal and job-related information. Administrator can also view the performance report of the staff.

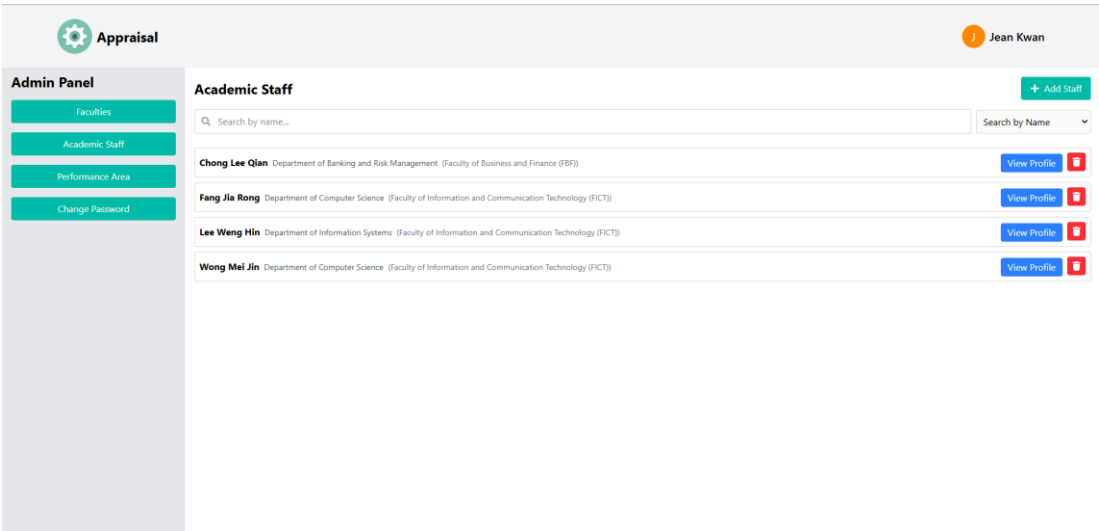


Figure 4.5.1 Academic Staff Page

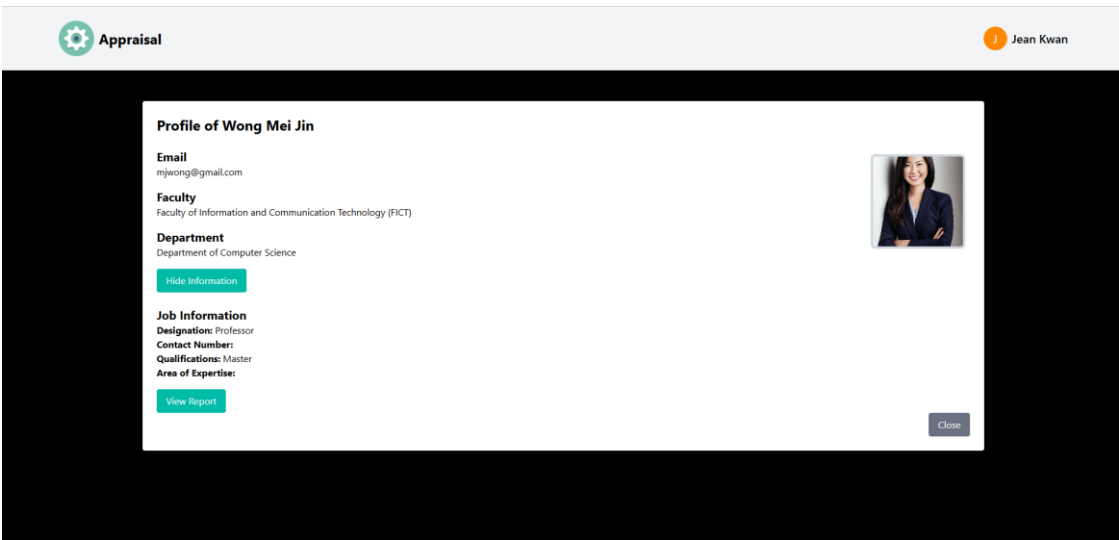


Figure 4.5.2 View Profile of Academic Staff

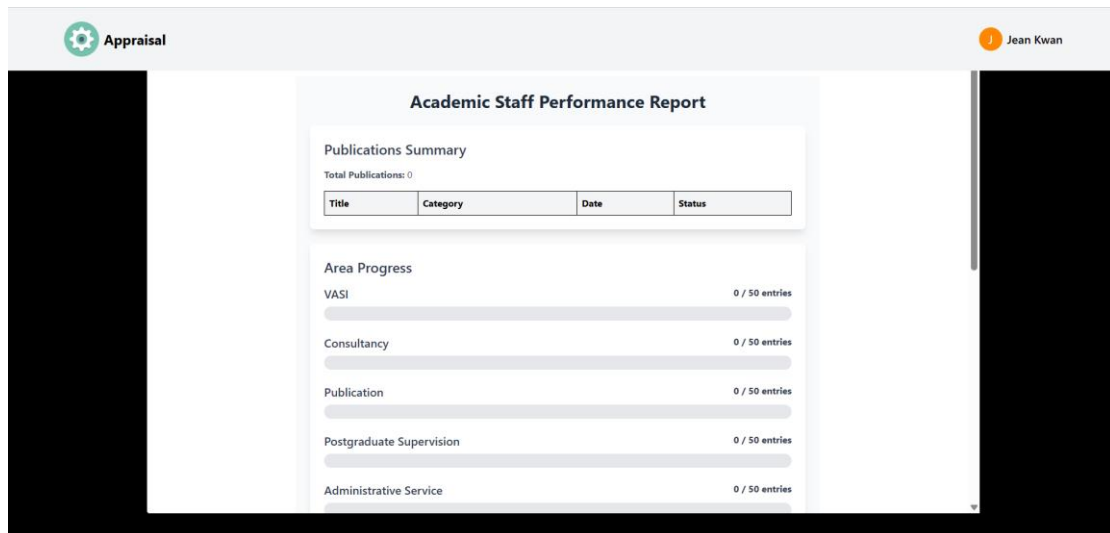


Figure 4.5.3 View Academic Staff Performance Report

4.6 Manage Performance Area Weightage

Different performance areas may contribute to different weightage, as not all performance areas are equally important. This weightage ensures that a major area will carry more weight in contributing to the overall scores. Administrator is allowed to modify the weightage of each performance area. This is due to the university's goals and priorities changing over time. Besides, the management may require the differences to focus on specific performance areas at particular times. Thus, modifiable weightage allows the system to evolve with these strategic changes.

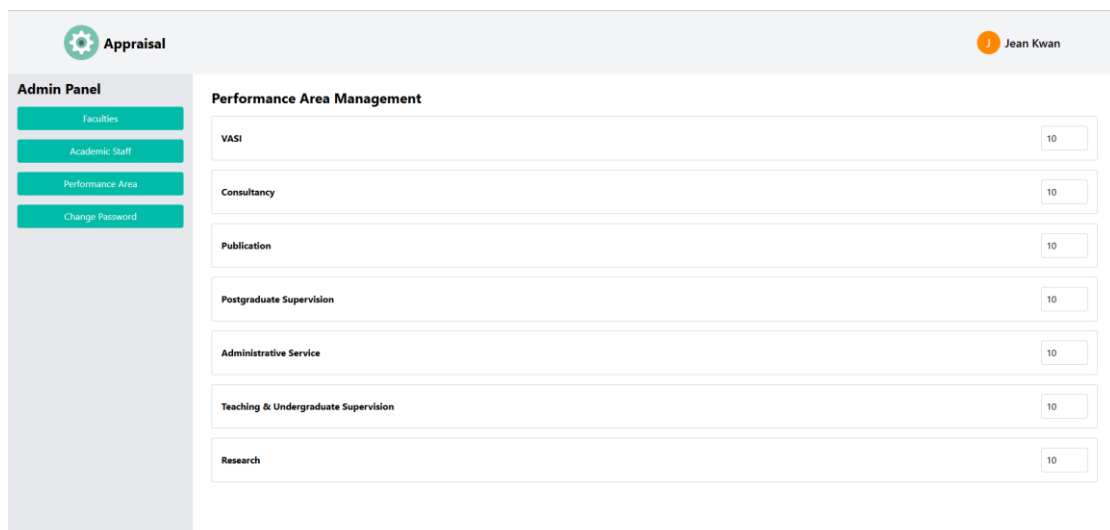


Figure 4.6.1 Performance Area Weightage Adjustment Page

4.7 View Performance Report

4.7.1 Staff Panel

To track the staff's achievement, staff can view their performance report by clicking on the performance reporting button from the side navigation bar. In the performance report, there are three parts in total. The first part is the summary of the staff's achievements. It shows the title, area, and the date of the achievement being added. The second and third parts are the progress of each area. In the second part, the user can view how many entries they have added to each area over the total entries, while the third part is a bar chart of the overall performance summary.

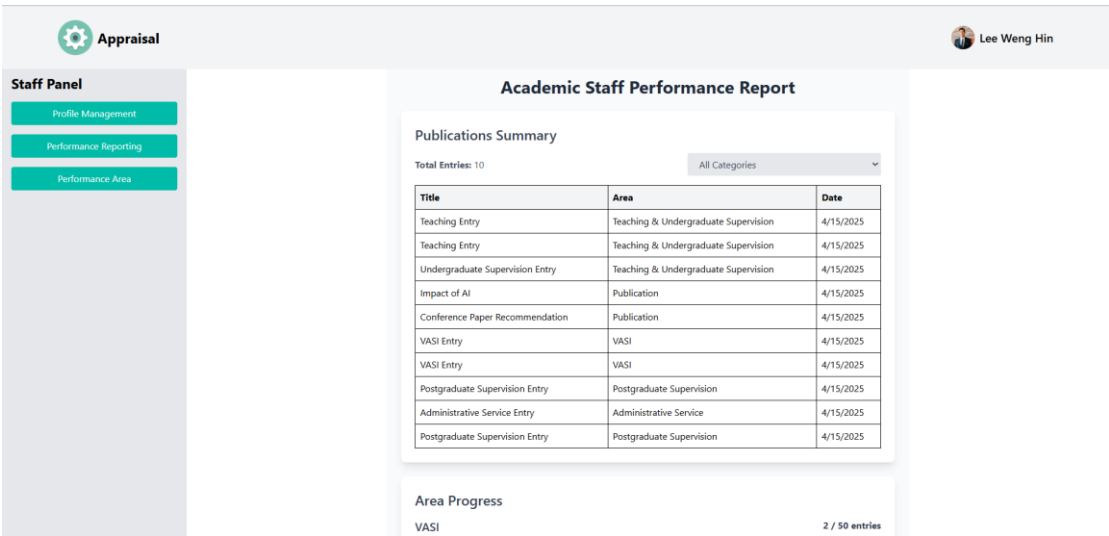


Figure 4.7.1.1 Staff Performance Report (Summary)

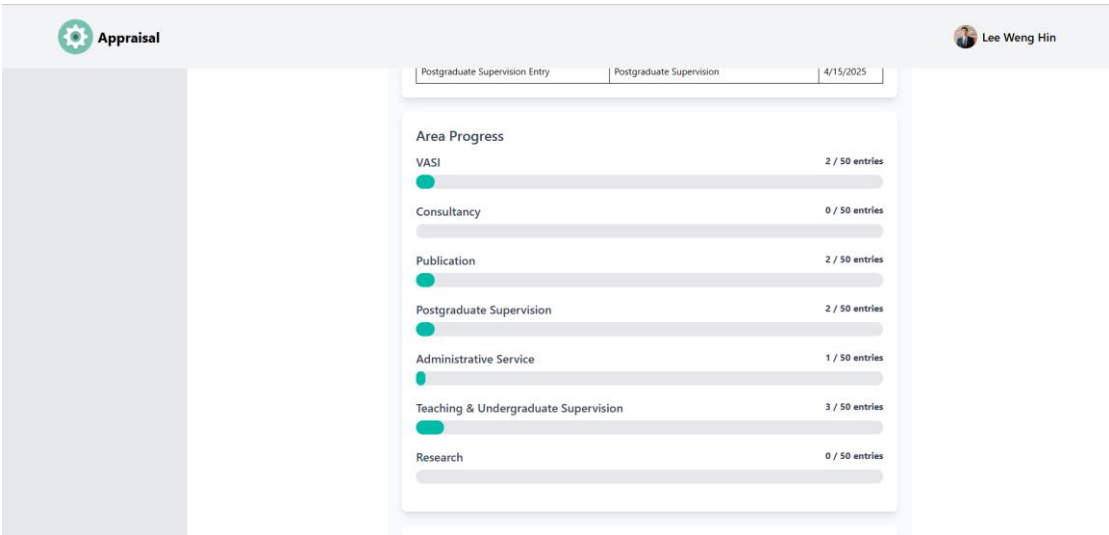


Figure 4.7.1.2 Staff Performance Report (Area Progress)

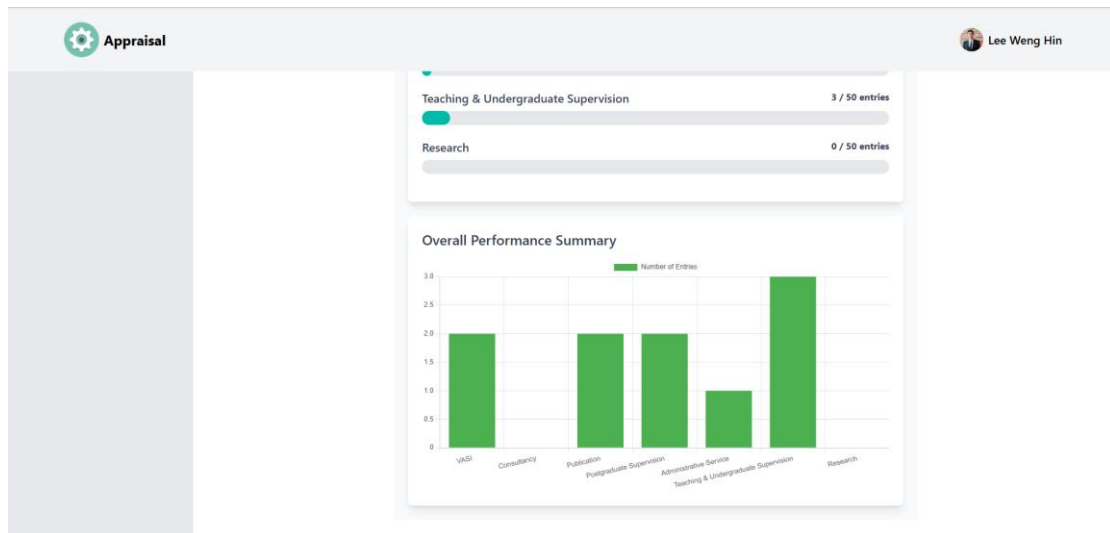


Figure 4.7.1.3 Staff Performance Report (Area Progress in Bar Chart)

4.7.2 Superior Panel

In the side navigation bar of the superior panel, there is an additional button, named view all staff. From there, the superior can also view the performance report of staff who work under them.

4.8 Add and View Performance Area Entry

By clicking the performance area button from the side navigation bar, the staff will be redirected to this page, where they can view the details of each achievement added. The staff can add a new performance entry by selecting the area from the drop-down list. Upon clicking the add entry button, a form modal will pop out. The form modal for each performance area varies as different information is needed. For some performance areas, the staff are required to upload a file as evidence of their achievement. This could prevent any fake achievement from being added.

Figure 4.8.1 Performance Entry Main Page

Figure 4.8.2 Form of Performance Entry (Publication)

4.9 Viewable Staff Page (Superior)

As mentioned before, a superior (staff with the assigned privilege) is able to view the records, including the profile and performance report of staff who work under them. This enables the superior to have a better understanding of the staff and be able to determine whether the performance of the staff has met the requirements. In the “Viewable Staff” page, to make searching for staff to be more easier when there are many staff under a superior, a search function is provided. With this, the superior can search the staff by their name. To view the staff, profile, the superior can click on the view profile button on the same row as each staff’s name. Basic information such as

CHAPTER 4 PRELIMINARY WORK

name, email, faculty and department of the staff will be displayed. To get more information like designation, qualification, and area of expertise, the superior can click on the more information button. Furthermore, a view report button is also provided, where the performance report of the staff will be displayed.

In addition, there is a view entries summary button on the “Viewable Staff” page. The staff name and entries of each performance area will be displayed in a table format. This provides a brief view of the progress of each staff. The superior can either sort the table by alphabetical order (by default) or by the number of total entries, either from low to high or from high to low.

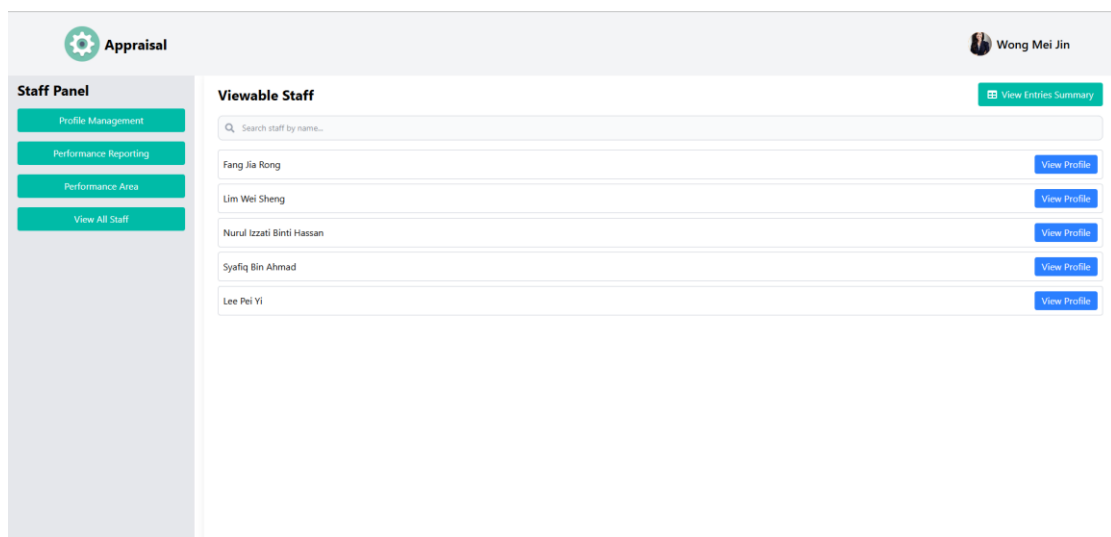


Figure 4.9.1 View All Staff Page (Superior)

The screenshot shows a 'Staff Entries Summary' table. The table has columns for Staff, Publication, Research, Teaching & Undergrad, Postgrad, VASI, Admin Service, Consultancy, and Total Entries. The data is as follows:

Staff	Publication	Research	Teaching & Undergrad	Postgrad	VASI	Admin Service	Consultancy	Total Entries
Aminah Binti Zulkiffi	0	1	1	0	0	0	0	2
Anjali A/P Manikam	0	0	0	0	0	0	1	1
Kavitha A/P Subramaniam	0	0	0	0	0	0	0	0
Lee Weng Hin	2	0	3	2	2	1	0	10
Tan Jia Hao	0	0	0	0	0	0	0	0

Figure 4.9.2 View Staff Entries Summary (Superior)

4.10 Summary

In conclusion, the preliminary work of this project includes the development of the login module, profile management module, report generation module, and administration module. Each module plays a crucial role in establishing the functionality of the employee appraisal system.

CHAPTER 5 CONCLUSION

The performance appraisal of employees is an important aspect of organizations, especially in educational institutions. This is because it offers constructive feedback and aids in the development of staff. However, traditional performance evaluation methods may face some limitations, such as difficulty in tracking staff's accomplishments and the manual evaluation process, which is time-consuming and inconsistent. These limitations have motivated the development of this employee appraisal system to enhance the evaluation flow.

The methodology used in developing this system is the RAD model, which enables flexible development and iterative enhancement. During Final Year Project 1, the core functionalities were implemented, including a login module, profile management module, administration module, and report generation module. Now, the superior can easily view information about their subordinates, such as the profile, the accomplishments added, and the progress of each performance area once access is granted by the administrator.

In short, the basic workflow for the employee appraisal system was developed during the Final Year Project 1. However, due to a lack of time, modules like the calendar module and the performance evaluation module were not implemented. Thus, the incomplete modules will be carried forward to Final Year Project 2 in order to achieve all the objectives of the project.

REFERENCES

REFERENCES

- [1] A. Hayes, “Performance Appraisals in the Workplace: Use, Types, Criticisms,” Investopedia, May 25, 2023. <https://www.investopedia.com/what-is-a-performance-appraisal-4586834#:~:text=The%20term%20%E2%80%9Cperformance%20appraisal%E2%80%9D%20refers>
- [2] C. Beveridge, “Performance Appraisal Systems: Everything You Need to Know,” 15Five, Jan. 25, 2024. <https://www.15five.com/blog/performance-appraisal-systems-everything-you-need-to-know/>
- [3] J. Terra, “What is a Performance Appraisal? Methods, Process, and Everything You Should Know,” Simplilearn.com, 2022. <https://www.simplilearn.com/what-is-performance-appraisal-methods-process-article>
- [4] O. W. Samuel, M. O. Omisore, and E. J. Atajeromavwo, “Online fuzzy based decision support system for human resource performance appraisal,” *Measurement*, vol. 55, pp. 452–461, Sep. 2014, doi: <https://doi.org/10.1016/j.measurement.2014.05.024>.
- [5] F. dwi Kartikasari, S. Limanto, and N. cahya Puspita, “Integrated Performance Appraisal System with Management by Objective Method,” *IEEE Xplore*, Oct. 01, 2022. <https://ieeexplore.ieee.org/abstract/document/10010037/figures#figures> (accessed May 10, 2023).
- [6] “Employee Performance Evaluation System in PHP/MySQLi with Source Code - Updated,” *SourceCodester*, 2020. <https://www.sourcecodester.com/php/14617/employee-performance-evaluation-system-phpmysqli-source-code.html>
- [7] People HR, “PeopleHR - Employee Performance Review,” *YouTube*, Jun. 13, 2014. <https://www.youtube.com/watch?v=Tpg7H4bEIRs> (accessed Aug. 31, 2024).
- [8] Inkrease BV, “Trakstar Demo (English),” *YouTube*, Feb. 13, 2013. <https://www.youtube.com/watch?v=kb1G-x2sqIQ> (accessed Aug. 31, 2024).

REFERENCES

- [9] “RAD (Rapid Application Development): Definition, Meaning and Benefits,”
kissflow.com. <https://kissflow.com/application-development/rad/rapid-application-development-rad-meaning/>

APPENDIX